A THREE DIMENSIONAL SOLUTION OF THE TRANSIENT FIELD PROBLEM USING ISOPARAMETRIC FINITE ELEMENTS

Girard Thomas Lew

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A THREE DIMENSIONAL SOLUTION OF THE
TRANSIENT FIELD PROBLEM USING
ISOPARAMETRIC FINITE ELEMENTS

by

Girard Thomas Lew

Thesis Advisors: D. Salinas
G. Cantin

December 1972

A Three Dimensional Solution of the Transient Field Problem Using Isoparametric Finite Elements

by

Girard Thomas Lew
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1960

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1972

ABSTRACT

An isoparametric finite element formulation solving a general form of the transient field equation is presented in this thesis. The formulation is developed for fields in three-dimensional Euclidean space. A FORTRAN IV computer program employing double precision arithmetic, compact storage techniques, and providing the option of several numerical integration methods and types of finite elements is presented. Data input may be in rectangular, cylindrical, or spherical coordinates and variations in time integration step size are permitted. Time dependent boundary conditions are not considered.

Comparisons of theoretical and computer solutions for a variety of test problems demonstrate close agreement. Instructions for use of the program are included.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure

LIST OF SYMBOLS AND NOTATION

A. NOTATION

$< >$	Row Vector
$[\ ]$	Square Matrix
$< >^T$	Transpose of a Row Vector
$[\ ]^T$	Transpose of a Square Matrix
$[\ ]^{-1}$	Inverse of a Square Matrix
$\{\ \}$	Column Vector
$[\ ]^e$ or $[\ ]^e$	Matrix or Vector at the Element Level
$\{\phi\}^i$	Vector at i^{th} Point in Time
$\{\dot{\phi}\}$	First Partial Derivative of $\{\phi\}$ with Respect to Time
$\{\ddot{\phi}\}$	Second Partial Derivative of $\{\phi\}$ with Respect to Time
$\int_v$ or $\int_{ve}$	Volume Integral
$\int_s$ or $\int_{se}$	Surface Integral
ds	Differential Area in Global Coordinate System
ds_e	Differential Area in Local Coordinate System

B. SYMBOLS

$[C]$ or $[C]^e$	Coefficient Matrix of $\{\dot{\phi}\}$ or $\{\dot{\phi}\}^e$
$\{F\}$ or $\{F\}^e$	Forcing Vector
$[G]$ or $[G]^e$	Coefficient Matrix of $\{\ddot{\phi}\}$ or $\{\ddot{\phi}\}^e$
$[H]$ or $[H]^e$	Coefficient Matrix of $\{\phi\}$ or $\{\phi\}^e$
$[J]$ or $[J_s]$	Jacobian Matrix of Volume or Surface

$\det[J]$ or $\det[J_s]$	Determinant of Jacobian Matrix of Volume or Surface
h	Time Step Size or Heat Transfer Coefficient
k_x, k_y, k_z	Material Property of [H] Matrix in x,y, or z Direction
k	Thermal Conductivity
ℓ_x, ℓ_y, ℓ_z	Direction Cosines
$<N>$	Shape Function in Space
$<M>$	Shape Function in Time
q	Loss Coefficient per Unit Area or Heat Flux per Unit ARea
Q	Generation per Unit Volume or Heat Generation per Unit Volume
r	Radius
r_o	Outside Radius
r_i	Inside Radius
t	Time
T	Temperature
α	Loss Coefficient associated with or Thermal Diffusivity
μ	Material Property of [C] Matrix
ρ	Material Property of [G] Matrix or Density
ν	Damping Factor

ACKNOWLEDGEMENTS

The author wishes to express his appreciation to Dr. Gilles Cantin, Professor of Mechanical Engineering, for his advice and guidance throughout the course of development of this text. Without his perseverant assistance this work could not have been accomplished.

The author is obligated to Professor Paul Pucci and Associate Professor Matthew Kelleher for their kind and thoughtful assistance in the preparation of test problems for this work and to Assistant Professor David Salinas for his assistance in the development of time integration procedures.

Finally the author wishes to thank his wife, Annette, for her understanding and encouragement throughout the course of this study.

I. INTRODUCTION

The 'quasi-harmonic' partial differential equation:

$$\frac{\partial}{\partial x}\left(k_x \frac{\partial \phi}{\partial x}\right) + \frac{\partial}{\partial y}\left(k_y \frac{\partial \phi}{\partial y}\right) + \frac{\partial}{\partial z}\left(k_z \frac{\partial \phi}{\partial z}\right) + \left(Q - \mu \frac{\partial \phi}{\partial t} - \rho \frac{\partial^2 \phi}{\partial t^2}\right) = 0 \qquad (1)$$

applies to a variety of physical problems [Ref. 10]. The Laplace equation is a typical and well known example of a specialized form of this equation. Since an analytical solution in closed form is seldom available for useful engineering problems, the development of a practical general purpose computerized solution method represents a useful addition to the analytical tools of the engineer.

This thesis describes a general purpose FORTRAN IV computer program using the isoparametric finite element approach to solve the field equation in general form. The program does not presently permit a time variation of boundary conditions or a functional variation of material properties.

The program allows selection of the linear, quadratic, or cubic isoparametric finite element with the option of specifying two through six Gauss points for numerical integration in space. Data input may be a rectangular, cylindrical, or spherical coordinates. Five time integration

routines are provided. Output is presented in a tabular form of node number, space coordinate, and functional value.

The structure of the program is modular in order to accommodate improvements and adjustments due to new developments.

This program was constructed and tested on an IBM-360/67 computer system with OS/360 release 18. Other systems and installations will require modifications.

II. FINITE ELEMENT FORMULATION

The 'quasi-harmonic' or field equation describing the behaviour of some physical quantity, ϕ, is:

$$\frac{\partial}{\partial x}\left(k_x \frac{\partial \phi}{\partial x}\right) + \frac{\partial}{\partial y}\left(k_y \frac{\partial \phi}{\partial y}\right) + \frac{\partial}{\partial z}\left(k_z \frac{\partial \phi}{\partial z}\right) + \left(Q - \mu \frac{\partial \phi}{\partial t} - \rho \frac{\partial^2 \phi}{\partial t^2}\right) = 0 \qquad (1)$$

Commonly imposed boundary conditions are of the forms:

$$\phi = \phi_B \qquad (2)$$

$$k_x \frac{\partial \phi}{\partial x} \ell_x + k_y \frac{\partial \phi}{\partial y} \ell_y + k_z \frac{\partial \phi}{\partial z} \ell_z + q + \alpha\phi = 0 \qquad (3)$$

Where ϕ_B represents the value of ϕ specified on the boundary; ℓ_x, ℓ_y, and ℓ_z are the direction cosines of the outward normal to the boundary surface; q is an outward flux per unit area; Q a generation per unit volume; and k_x, k_y, k_z, μ, and ρ are material properties. Q, k_x, k_y, k_z, μ, and ρ may be functions of space and time.

Reference 10 thoroughly develops the concept of a finite element and the process leading to a discretized form of equation (1) and the boundary conditions given by equations (2) and (3). If the scalar variable, ϕ, is described by a shape function, <N>, and nodal values of the variable, $\{\phi\}$, such that:

$$\phi = \langle N \rangle \{\phi\} \qquad (4)$$

the result is:

$$[H]\{\phi\} + [C]\{\dot{\phi}\} + [G]\{\ddot{\phi}\} + \{F\} = 0 \tag{5}$$

[H], [C], and [G] are symmetric matrices defined, on the element level, as:

$$[C]^e = \int_{v_e} \langle N \rangle^T \mu \langle N \rangle \, dx \, dy \, dz \tag{6}$$

$$[G]^e = \int_{v_e} \langle N \rangle^T \rho \langle N \rangle \, dx \, dy \, dz \tag{7}$$

$$[H]^e = \int_{v_e} \left(k_x \left\langle \frac{\partial N}{\partial x} \right\rangle^T \left\langle \frac{\partial N}{\partial x} \right\rangle + k_y \left\langle \frac{\partial N}{\partial y} \right\rangle \left\langle \frac{\partial N}{\partial y} \right\rangle + k_z \left\langle \frac{\partial N}{\partial z} \right\rangle^T \left\langle \frac{\partial N}{\partial z} \right\rangle \right) dx \, dy \, dz \tag{8}$$

The load vector describing boundary conditions is:

$$\{F\}^e = -\int_{v_e} Q \langle N \rangle^T \, dxdydz + \int_{s_e} q \langle N \rangle^T \, ds_e + \left(\int_{s_e} \langle N \rangle^T \alpha \langle N \rangle \, ds \right) \{\phi\}^e \tag{9}$$

where the integrals are performed only over the surface or volume where the prescribed boundary condition applies.

Element matrices and the load vector are assembled by standard methods to form equation (5) which is the discretized form of equation (1). Adding the load vectors given by equation (9) to the rest of the nodal intensities guarantee the satisfaction of the boundary conditions given by equation (3). Finally, the boundary condition of equation (2) can be applied to the assembled set of equations.

The finite element technique models a continuum as an assembly of substructures; thus, for an element of small size, the parameters k_x, k_y, k_z, μ, ρ, α, Q, and q may be considered constant and the integrations required by equations (6), (7), (8), and (9) reduced to an automated numerical technique.

A. ISOPARAMETRIC FINITE ELEMENTS

The linear, quadratic, and cubic isoparametric shape functions are utilized in this development. The resulting elements contain 8, 20, and 32 nodal points respectively. A full development of the concept and properties of this family of finite elements is contained in Ref. 10; only the necessary results will be repeated here.

A "local" non-dimensional system of reference (ξ,η,ζ) is used for convenience of calculation. Results are transformed into the "global" cartesian (x,y,z) system of reference prior to final assembly of problem matrices and vectors. The shape functions, $N_i(\xi,\eta,\zeta)$, for the element types used in this development are given in Ref. 10 and need not be repeated here. The nodal numbering conventions employed for each element type are shown in Figures 1, 2, and 3.

Important results relating the shape function to the local and global reference systems [Ref. 10] are:

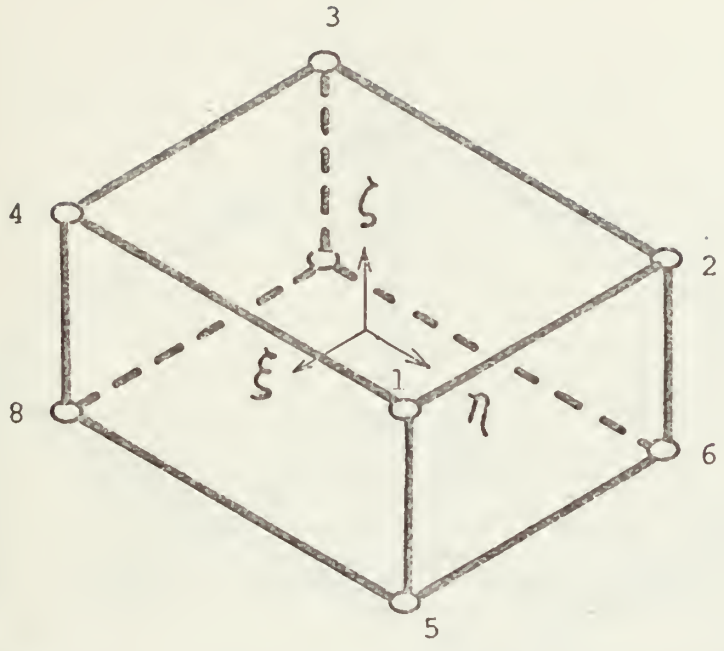


Figure 1. Linear Isoparametric Element.

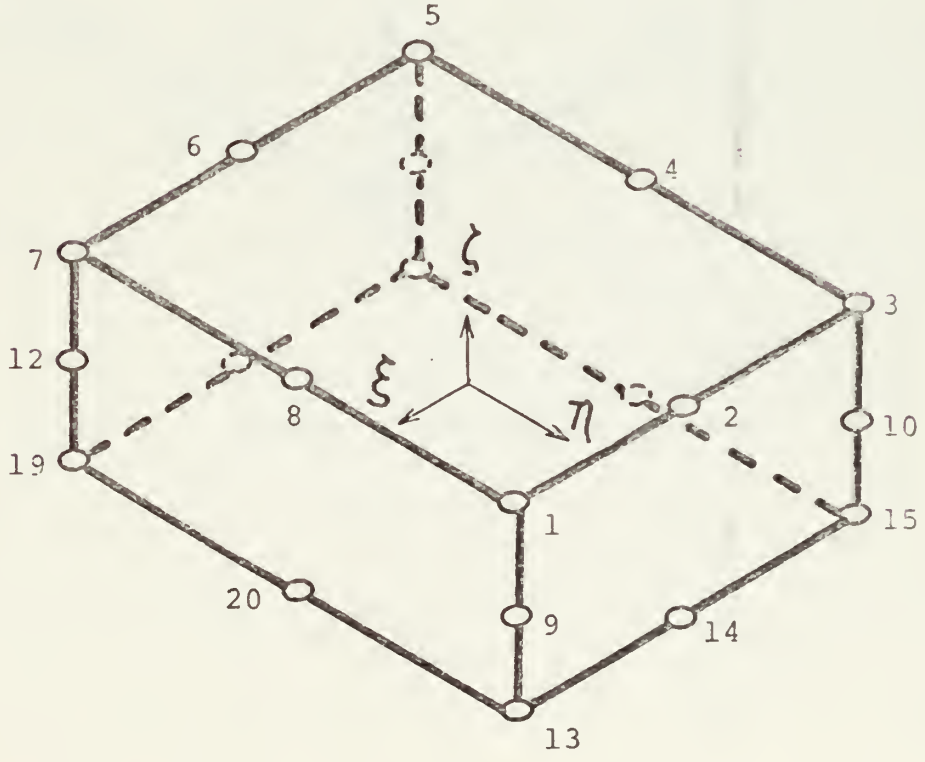


Figure 2. Quadratic Isoparametric Element.

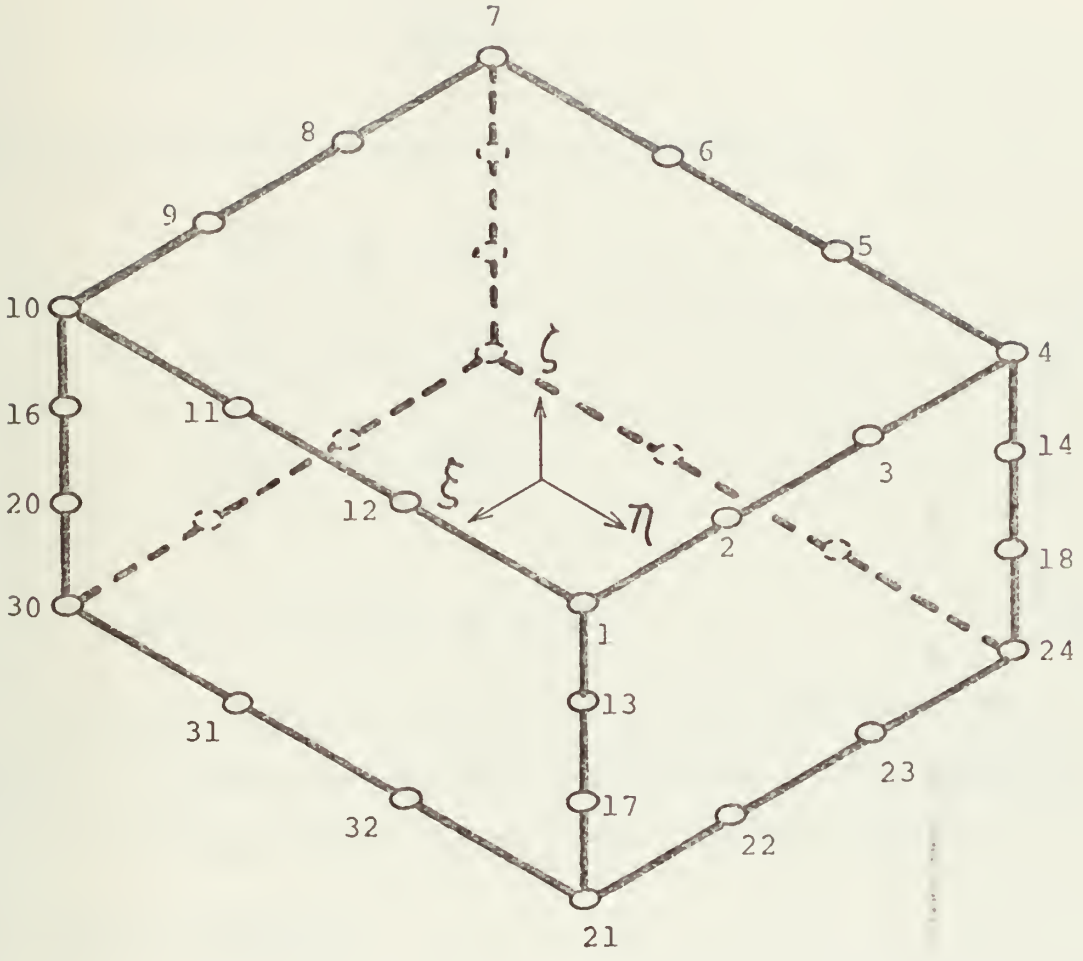


Figure 3. Cubic Isoparametric Element.

$$\begin{Bmatrix} \frac{\partial N_i}{\partial \xi} \\ \frac{N_i}{\partial \eta} \\ \frac{\partial N_i}{\partial \zeta} \end{Bmatrix} = [J] \begin{Bmatrix} \frac{\partial N_i}{\partial x} \\ \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial z} \end{Bmatrix} \tag{10}$$

where the Jacobian matrix, [J], is defined as

$$[J] = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix} \tag{11}$$

and

$$dxdydz = \det [J] \, d\xi \, d\eta \, d\zeta \tag{12}$$

The partial derivatives of the shape function with respect to global coordinates are recovered from equation (10) by applying the inverse of the Jacobian matrix.

B. MATRIX FORMATION

Assuming constant material properties within an element and applying equations (10), (11), and (12) to the matrix and vector formulations of equations (6), (7), (8) and (9) reduces the form of the integrals to:

$$\int \langle N \rangle^T \langle N \rangle \det [J] \, d\xi d\eta d\zeta$$

$$\int \langle \frac{\partial N}{\partial x} \rangle^T \langle \frac{\partial N}{\partial x} \rangle \det [J] \, d\xi d\eta d\zeta + \ldots$$

$$\int \langle N \rangle^T \det [J] \, d\xi d\eta d\zeta$$

$$\int \langle N \rangle^T \det [J_s] \, ds_e$$

$$\int \langle N \rangle^T \langle N \rangle \det [J_s] \, ds_e$$

where the integrations are performed in the local nondimensional coordinate system. The forms of equations (6), (7), (8), and (9) become ideally suited for a Gaussian numerical integration procedure.

C. BOUNDARY CONDITIONS

The specification of a value of the function ϕ along a given boundary as in equation (2) is straightforward and requires no comment.

Boundary specifications of equation (3) appear in the expression for a forcing vector given by equation (9). The first term of equation (9) containing Q represents a generation rate per unit volume; the second term containing q represents a flux per unit area; and the third term containing α represents a surface loss coefficient per unit area. The terms containing q and Q contribute to a forcing vector, {F}, while the term containing α results in a correction to the corresponding element [H] matrix.

If, in equation (5), q and α are zero and k_x, k_y, and k_z are equal the boundary condition reduces to the requirement that the normal derivative be equal to zero. This condition is automatically imposed by equation (5) in the absence of any other boundary condition specification.

Arpaci [Ref. 1] gives a number of boundary conditions applicable to the heat problem. Of these the Radiation,

Interface in Motion, and Change of Phase are not included in this development.

Consider an element located at the surface of a body in space. From Figures 1, 2, and 3 observe that a corner node may be common to one, two, or three boundary surfaces per element and that a mid side node is common to one or two boundary surfaces per element. Each boundary surface may be subject to a different boundary condition specification. If two or more of the element surfaces have different boundary conditions imposed, a unique situation arises if the boundary specifications include conditions on ϕ and q or α. In this case the boundary conditions must be specified in order. First, all boundary conditions involving q and α are formed and merged into the overall system of equations; then the boundary conditions involving ϕ are applied.

D. SOLUTION TECHNIQUES

For a fixed point in time and for steady state problems, equation (5) reduces to the form

$$[A]\{\phi\} = \{B\} \tag{13}$$

where [A] is a n x n square matrix. The vector {B} is fully determined in all cases except where a nodal value of ϕ is specified. If a nodal value, ϕ_i, is specified the corresponding forcing vector term, B_i, will contain the unknown quantity of the equation.

1. Systems of Linear Equations

Equation (13) may be solved by standard methods of linear algebra. The method employed here to sort the known and unknown quantities of equation (13) was developed by Irons [Ref. 6] and has the advantages of preserving the symmetry of matrix [A] and permitting the determination of all unknown quantities of the system of equations.

After applying the Irons [Ref. 6] modifications, solution of equation (13)' is accomplished by the reduction of matrix [A] into the product of lower unit triangular, diagonal, and upper triangular matrices such that:

$$[A] = [L]\ [D]\ [L]^T$$

Unknown quantities are found by performing a forward substitution followed by a backward substitution [Ref. 13].

2. Steady State

In steady state, equation (5) readily reduces to the form of equation (13) and is solved by the technique described previously.

3. First Order in Time

At fixed points in time, equation (5) reduces to the form of equation (13). The numerical integration schemes employed are straightforward and are summarized below. The initial value of $\{\phi\}$ is assumed known and time step size is denoted by h.

a. Euler I Method

This integration scheme is well documented by Crandall [Ref. 3]. The result, for the i^{th} point in time, is:

$$[C]\{\dot{\phi}\}^{i-1} = -[H]\{\phi\}^{i-1} - \{F\} \tag{15a}$$

$$\{\phi\}^{i} = \{\phi\}^{i-1} + h\,\{\dot{\phi}\}^{i-1} \tag{15b}$$

b. Trapezoidal Rule Method

This approach to time integration is documented by Ref. 3 and Ref. 11. At mid interval

$$\{\phi\}^{i+\frac{1}{2}} = \frac{\{\phi\}^{i} + \{\phi\}^{i+1}}{2}$$

$$\{\dot{\phi}\}^{i+\frac{1}{2}} = \frac{\{\phi\}^{i+1} - \{\phi\}^{i}}{h}$$

When applied to equation (5), the resulting recursion formula is:

$$\left([H] + \frac{1}{h}[C]\right)\{\phi\}^{i+1} = \left(-[H] + \frac{1}{h}[C]\right)\{\phi\}^{i} - 2\{F\} \tag{16}$$

c. Linear Time Shape Function Method

Reference 10 develops the following approach to time integration. Assume that $\{\phi\}$ may be considered as a product of the nodal values of $\{\phi\}$ at discrete times and a time shape function, $M(t)$, such that

$$\langle M \rangle = \langle M_o(t), M_1(t) \rangle$$

where, for a linear variation:

$$M_o = \frac{(h-t)}{h}$$

and

$$M_1 = t/h$$

the result is:

$$\{\phi\} = \langle M_o, M_1 \rangle \begin{Bmatrix} \{\phi\}^{i-1} \\ \\ \{\phi\}^{i} \end{Bmatrix}$$

Applying this relationship and the Galerkin process described in Ref. 10 results in the formula:

$$(\frac{2}{3}[H] + \frac{1}{h}[C])\{\phi\}^{i} = -(\frac{1}{3}[H] - \frac{1}{h}[C])\{\phi\}^{i-1} - \{F\} \quad (17)$$

4. Second Order in Time

At fixed points in time, equation (5) reduces to the form of equation (13). The integration methods employed are summarized below. Initial values of $\{\phi\}$ and $\{\dot{\phi}\}$ are assumed to be known. Time step size is denoted by h.

a. Euler I Method

The Euler I integration method is well known and documented by many standard sources such as Ref. 3. The result, when applied to equation (5) is:

$$[G]\{\ddot{\phi}\}^{i-1} = -[H]\{\phi\}^{i-1} - [C]\{\dot{\phi}\}^{i-1} - \{F\} \quad (18a)$$

$$\{\phi\}^{i} = \{\phi\}^{i-1} + h\{\dot{\phi}\}^{i-1} \quad (18b)$$

$$\{\dot{\phi}\}^{i} = \{\dot{\phi}\}^{i-1} + h\{\ddot{\phi}\}^{i-1} \quad (18c)$$

In the event that the [C] matrix equals zero these equations remain valid.

b. Finite Difference Method

Finite difference approximations are developed in Ref. 3. A four point backward difference approximation for $\{\ddot{\phi}\}$ and a three point backward difference approximation

for $\{\dot{\phi}\}$ have truncation errors of the same order of magnitude, h^2, and are:

$$\{\ddot{\phi}\}^i = \frac{1}{h^2}\left(-\{\phi\}^{i-3} + 4\{\phi\}^{i-2} - 5\{\phi\}^{i-1} + 2\{\phi\}^i\right) \quad (19a)$$

$$\{\dot{\phi}\}^i = \frac{1}{h}\left(\frac{1}{2}\{\phi\}^{i-2} - 2\{\phi\}^{i-1} + \frac{2}{3}\{\phi\}^i\right) \quad (19b)$$

Define

$$\{\phi^*\} = \frac{1}{h^2}\left(-\{\phi\}^{i-3} + 4\{\phi\}^{i-2} - 5\{\phi\}^{i-1}\right) \quad (19c)$$

and

$$\{\phi^{**}\} = \frac{1}{h}\left(\frac{1}{2}\{\phi\}^{i-2} - 2\{\phi\}^{i-1}\right) \quad (19d)$$

Equations (19a) and (19b) when applied to equation (5) give the following recursion formula

$$\left([H] + \frac{2}{h^2}[G] + \frac{3}{2h}[C]\right)\{\phi\}^i = - \{F\} - [G]\{\phi^*\} - [C]\{\phi^{**}\} \quad (20)$$

if [C] is not equal to zero and

$$\left([H] + \frac{2}{h^2}[G]\right)\{\phi\}^i = - \{F\} - [G]\{\phi^*\} \quad (21)$$

if [C] equals zero.

Equation (21) is known as the Houbolt method and is shown by Johnson [Ref. 7] to be unconditionally stable.

Three starting values of $\{\phi\}$ must be obtained by independent means such as the application of the Euler I method or a Taylor series expansion.

III. DESCRIPTION OF THE PROGRAM

A functional block diagram of the program is presented in Figures 4 through 9. A modular structure was adopted in order to permit simple modifications to accommodate future improvements and the adaptation of the program for special purposes.

The following paragraphs give a brief description of the function and operation of the program. Appendices A through D discuss the preparation of input data, list the important computer program nomenclature, and list the program. A full understanding of the operation of this program can only be achieved by a study of Appendices A through D and an understanding of the principles and solution techniques employed.

A. ASSUMPTIONS AND RESTRICTIONS

The key assumptions and restrictions which impose restrictions in the type of problems that can be solved by the program are:

1. Material properties k_x, k_y, k_z, ρ, μ, and α and problem parameters q and Q are constant functions of time and space throughout the element and are independent of the function ϕ.

2. Boundary values of ϕ are not functions of time.

3. The principal axes of each element align with the principal axes of the global coordinate system and are fixed

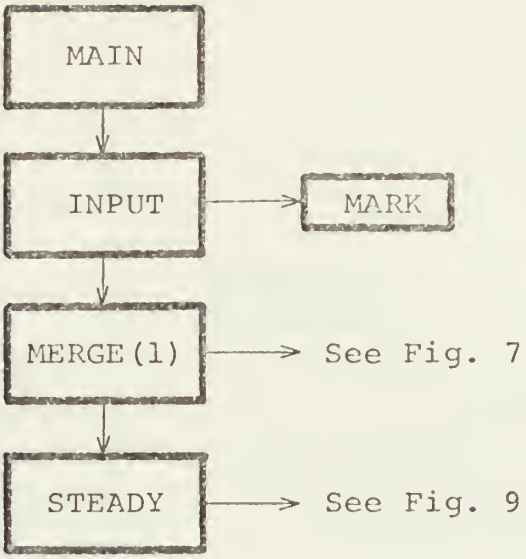


Figure 4. Subroutine Calling Sequence for Steady State Problem.

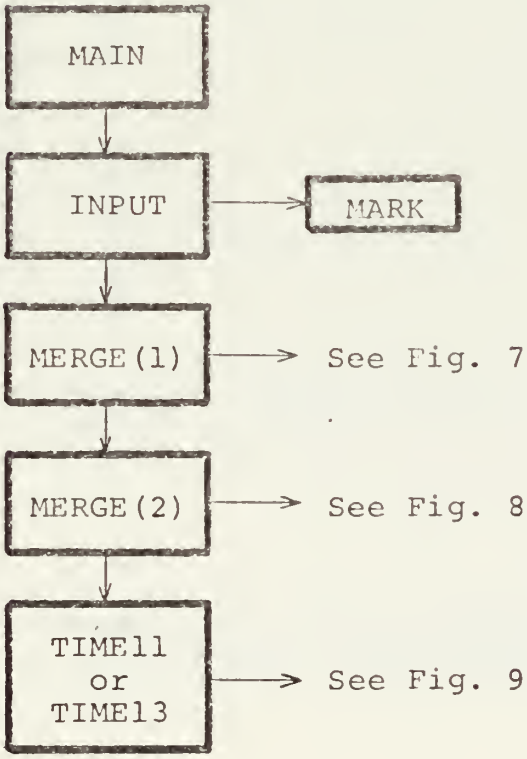
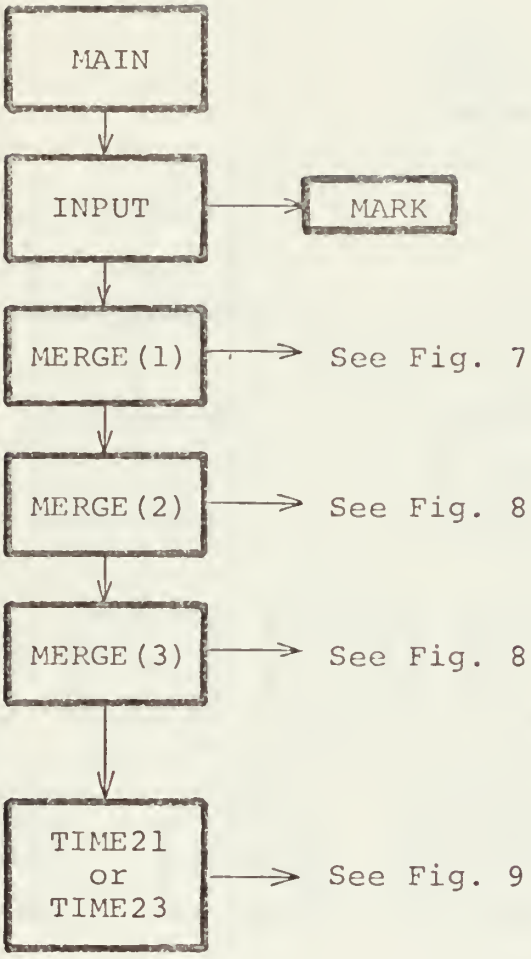
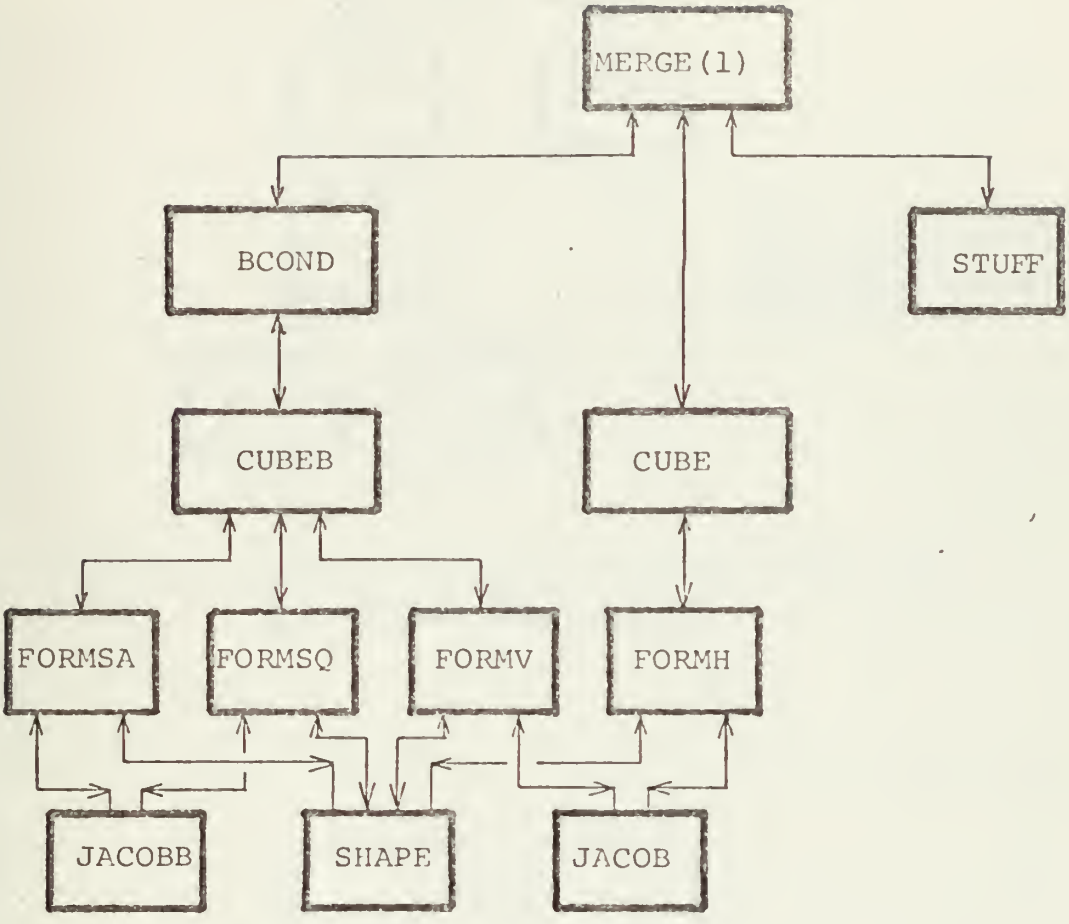


Figure 5. Subroutine Calling Sequence for First Order Time Problem.



NOTE: MERGE(2) is not called if the [C] matrix is absent

Figure 6. Subroutine Calling Sequence for Second Order Time Problem.



NOTE: STUFF not called for the Steady State Problem

Figure 7. Assembly Routines for the Formation of Matrix [H].

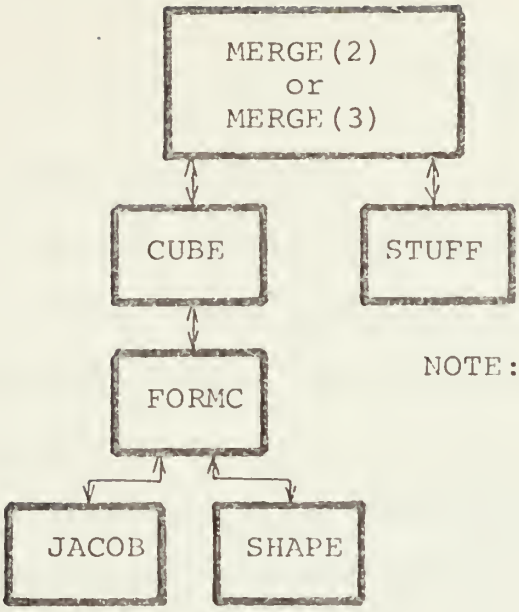
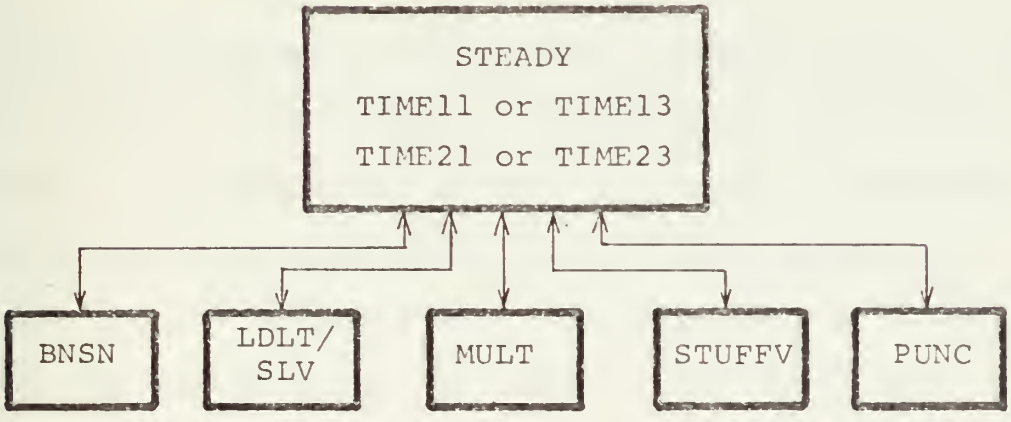


Figure 8. Assembly Routines for the Formation of Matrices [C] and [G].



NOTE: (1) STUFF called by TIME23 only

(2) MULT not called by STEADY

Figure 9. Service Routines Used by Solution Routines.

in direction throughout the element.

These limitations will be discussed in Section V.

B. GENERAL PROGRAM ORGANIZATION

The program may be divided into the three major independent functions: reading of input data, matrix formation, and actual problem solution. This organization is shown in Figures 4 through 9.

All matrices involved in the problem are symmetrical. At the element level the complete matrix is formed; however, formation of the master matrices [H], [C], and [G] and problem solution utilize a standard compressed storage mode to conserve computer core. The technique consists of placing the diagonal elements of each row of the matrix into the first column and retaining only the upper elements of the matrix.

Reading of necessary input data is accomplished by subroutine INPUT. Preparation of input data is discussed in Appendix B. An echo check of all input data is provided.

Formation of the required matrices and vectors is controlled by subroutine MERGE. Matrices [H], [C], and [G] are formed sequentially. This permits output of the completed matrix to an external storage device and conserves core storage. Boundary condition calculations are performed during formation of the [H] matrix. Associated subroutines include CUBE, FORM, and JACOB.

The transient solution subroutines are similar in construction and differ only in the integration procedure employed. Time integration step size changes and data output are controlled by the solution subroutines. Certain functions common to all solution subroutines are performed by a set of service subroutines which include LDLT, BNSN, STUFF, PUNC, and MULT.

C. SUBROUTINES

The MAIN program is simply a control segment that serves to call INPUT; MERGE(1), MERGE(2), or MERGE(3) in the proper order and as required; and the applicable solution subroutine STEADY, TIME11, TIME13, TIME21, or TIME23. No calculations are performed in the MAIN program

1. Subroutine INPUT

This subroutine reads all input data and provides the appropriate echo check. The number of nodes per element is determined from the element type specification and, if required, cylindrical or spherical coordinates are transformed into rectangular coordinates by calling subroutine MARK.

2. Subroutine MERGE(IMTRX)

This subroutine forms the master matrix specified by the calling argument, IMTRX, by assembling each element matrix into the master matrix. A call to subroutine CUBE returns the desired element matrix. IMTRX=1 corresponds to the [H] matrix, IMTRX=2 corresponds to the [C] matrix, and

IMTRX=3 corresponds to the [G] matrix. Subroutine BCOND is called during formation of the [H] matrix and, if required by the solution subroutine selected, the [H] or [C] matrices are placed into the appropriate storage location after formation.

3. Subroutine BCOND

This subroutine forms the element vectors and matrices required by the appropriate boundary condition specified and assembles the resulting vector or modification matrix into the forcing vector, {F}, or the corresponding [H] matrix. Formation at the element level is accomplished by a call to the appropriate entry point of subroutine CUBE.

4. Subroutine STEADY

This subroutine solves the steady state problem. After the application of appropriate nodal boundary conditions, a call to subroutine LDLT and entry point SLV produces the solution. Data output is a printout of the $\{\phi\}$ and {F} vectors and punched card output of the $\{\phi\}$ vector if requested.

5. Subroutine TIME11

This subroutine employs the Euler I time integration method, equations (15a) and (15b), to solve the first order time problem. One call to subroutine LDLT is required; subsequent calls to entry point SLV produce the desired solution. Data output is a tabular presentation of the vector at specified points in time with punched card output of the $\{\phi\}$ vector provided if requested.

6. Subroutine TIME13

This subroutine employs the "Linear Time Shape Function" as the time integration method (KINT=3), equation (17), or the Trapezoidal Rule time integration method (KINT=2), equation (16), to solve the first order time problem. External core storage devices are utilized to store the [H] and [C] matrices in order to permit restarting the problem at each change in time integration step size. One call to subroutine LDLT is required for each change in time integration step size with the required solution obtained by sequential calls to entry point SLV. Data output is the same as for TIME11.

7. Subroutine TIME21

This subroutine employs the Euler I time integration method, equations (18a), (18b), and (18c), to solve the second order time problem. Operation of this subroutine is identical to the operation of TIME11 except for the inclusion of the vector $\{\dot{\phi}\}$ in the data output.

8. Subroutine TIME23

This subroutine employs a Four Point Backward Finite Difference (Houbolt) time integration method, equations (20) and (21), to solve the second order time problem. Starting values are obtained by a five term Taylor Series expansion. Operation of this subroutine is similar to the operation of subroutine TIME13 employing external core storage devices and saving the values of $\{\phi\}$ required to restart the problem after a change in time integration step size. Data output is the same as for TIME21.

9. Subroutine CUBE(NPEL,NGP,ITYPE)

This subroutine performs the Gaussian numerical integration required for the formation of the various element matrices. Entry arguments are the number of nodes per element, number of Gauss points specified for the integration process, and the type of integration to be performed. Gauss points and weighting factors were obtained from Ref. 9. A selection of two through six Gauss points with a default option of three Gauss points is provided. For each specified Gaussian integration point, the numerical evaluation of the integrand is determined by calling subroutine FORMC at the appropriate entry point and applying the proper weight factor to the result prior to summation.

The specialized volume and surface integrals required for boundary condition calculations are obtained by entering this subroutine at entry point CUBEB.

10. Subroutine FORMC(XI,ETA,ZETA,NPEL)

This subroutine performs the products of $\langle N \rangle^T \langle N \rangle$, $\langle \frac{\partial N}{\partial x} \rangle^T \langle \frac{\partial N}{\partial x} \rangle$, etc. required at each point in local coordinate space for the numerical integration process. Values of the shape function and partial derivatives of the shape function with respect to the local coordinate system are determined by a call to subroutine SHAPE. Calling subroutine JACOB returns the required values of [J], $[J]^{-1}$, and Det[J]. Necessary entry arguments are the local coordinates of the point in space under consideration and the number of points per element. Various entry points are employed to accommodate

the specialized calculations required in the formation of each element matrix or vector and volume or area integrations.

11. Subroutine SHAPE(XI,ETA,ZETA,NPEL)

This subroutine forms the shape function and the partial derivative of the shape function with respect to the local coordinate system at the point in local coordinate space specified by the entry argument. The type of element to be considered is determined by the specification of the number of points per element in the calling argument.

12. Subroutine JACOB(NPEL,AJ,DCOL,CORD,AJI,DTJ)

This subroutine evaluates the Jacobian matrix, determines the inverse of the Jacobian matrix and the determinant of the Jacobian. The number of points per element, global coordinates of the node points, and shape function partial derivatives with respect to the local system of reference are required entry arguments. [J] is returned in AJ, $[J]^{-1}$ in AJI, and det[J] in DTJ.

The specialized calculations required for a surface integration are accomplished by entering this subroutine at entry point JACOBB.

13. Subroutine LDLT(A,N,M,ASN)

This subroutine reduces the matrix [A] into the products of lower unit triangular, diagonal, and upper triangular matrices, $[L]\ [D]\ [L]^T$, required by the process of solving the linear system of equations described by $[A]\ \{x\} = \{B\}$. N and M are the number of equations and half band width respectively. ASN is an arbitrarily small number;

numbers less than ASN squared are considered to be zero. Entry point SLV(Z,B,N,M) performs the forward substitution and backward substitution required to solve the system of equations.

14. Subroutine MARK(C,K,M)

This subroutine converts coordinates given in a cylindrical or spherical system of reference to a rectangular system of reference. C is a matrix containing the specified coordinates, K is a code to indicate the system of reference employed, and M is the number of coordinates specified.

15. Subroutine BNSN(A,N,APZRO,ABIGN)

This subroutine determines the arbitrarily small number required by subroutine LDLT and the arbitrarily large number required by the equation solving process. APZRO is determined from the matrix [A] by finding the average value of the diagonal elements and dividing the result by 10^{15}. ABIGN is 10^{30} x ASN. This provides the numerical spread necessary for the IBM/360 computer installation employed for this program development.

16. Subroutine MULT(A,B,C,N,M)

This subroutine performs the matrix multiplication [A] {B} = {C} in a compressed storage form. This multiplication is required by the various time integration subroutines. N and M are the number of equations and half band width respectively.

17. Subroutine STUFF(A,B,N,M)

This subroutine places the matrix [A] into matrix [B]; or entry STUFFV(AA,BB,N,MC) places the vector {B} into the specified column, MC, of matrix [AA].

18. Subroutine PUNC(A,B,N,K)

This subroutine produces the punched output requested by the solution subroutines STEADY, TIME11, TIME13, TIME21, or TIME23.

IV. TEST PROBLEMS

The test problems employed here are designed to test and verify the operation of the computer program. The four categories of tests performed were assembly, steady state, first order in time, and second order in time problems.

The assembly tests served to verify the calculations and integrations required to form the problem matrices and vectors on the element level. Steady state problems were used to verify the matrix assembly process, linear equation solution process, and the validity of boundary condition formulations. Transient problems of first and second order in time served to test the various solution subroutines utilized.

A. ASSEMBLY TESTS

A rectangular region in space provides a convenient means of verifying element matrix and vector assembly procedures. The calculation of all problem matrices and vectors is straightforward and element calculations of the [H], [C], and [G] matrices were verified in this manner. Similarly, the basic assembly processes of subroutines MERGE and BCOND are readily verified.

The integrations required by equations (6), (7), (8) and (9) are performed by a Gaussian numerical integration scheme. In the Gauss procedure, n sampling points will exactly integrate a polynomial of degree (2n-1). The integrations necessary may involve polynomials of degree 2 through 14 in

ıny given direction of the local coordinate system. Poly-
ıomial degree will depend on the shape of the element
elected. For practical applications, a selection of two
hrough six Gauss points is sufficient. For a rectangular
hape the linear quadratic, and cubic elements are exactly
ntegrated by 2, 3, and 4 Gauss points respectively. Veri-
ication of the number of Gauss points required to exactly
ntegrate elements of other shapes was not considered.

3. STEADY STATE PROBLEMS

1. Axial Temperature Distribution in a Cylinder

Consider a cylinder of the geometry specified in
igure 10. If the sides are perfectly insulated a linear
emperature distribution results when end conditions are
specified [Refs. 1 and 5]. Specify the thermal conductivity,
, as 20 BTU/hr/ft/°F and divide the region into two finite
lements. The top surface is subjected to an ambient tem-
erature of 100°F.

If the bottom of the cylinder is subjected to a
eat flux, q, of 500 BTU/hr/ft^2 or maintained at 300°F with
heat transfer coefficient, α, of 5 BTU/hr/ft^2/°F identical
emperature distributions result. Theoretical and finite
lement solutions are shown in Table I.

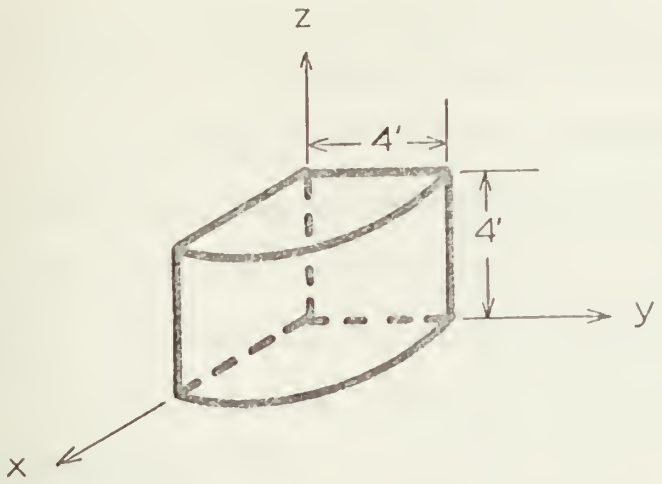


Figure 10. Axial Cylinder Problem Geometry.

TABLE I

AXIAL TEMPERATURE DISTRIBUTION IN A CYLINDER

Distance from Bottom(ft)	Temperature (°F)		
	Theoretical Solution	2 Quadratic Element Model	2 Cubic Element Model
0.00	200.00	200.00	200.00
0.67	183.33		183.33
1.00	175.00	175.00	
1.33	166.67		166.67
2.00	150.00	150.00	150.00
2.67	133.33		133.33
3.00	125.00	125.00	
3.33	116.67		116.67
4.00	100.00	100.00	100.00

2. Radial Temperature Distribution in a Cylinder

Consider a hollow cylinder of infinite length. A segment of such a cylinder is shown in Figure 11. Heat is lost by convection through the outer wall. Consider the case of the interior surface maintained at a constant temperature or the case of a specified heat generation within the cylinder with the interior surface perfectly insulated. Assumed problem parameters are:

a.	Inside radius (r_i)	3 in.
b.	Outside radius (r_o)	9 in.
c.	Heat transfer coefficient (α)	36 BTU/hr/ft^2/°F
d.	Thermal conductivity (k)	6 BTU/hr/ft/°F
e.	Ambient temperature	70°F
f.	Inside wall temperature	1500 °F
g.	Heat generation rate (Q)	432 BTU/hr/ft^3

Theoretical solutions to this problem are readily obtained by methods outlined in Refs. 1 and 5. Comparisons of solutions obtained by finite element models and theoretical means are given in Tables III and IV.

A finite element model of this problem is constructed by assembling elements radially to form a segment of the cylinder. Since the problem is symmetric and no heat is conducted perpendicular to any radius, any convenient specification of element arc width, θ, and axial thickness, L, may be used. A small element arc is desirable since the isoparametric finite element used will provide a better approximation to a true circular arc. A large arc width

will introduce some distortion in the temperature distribution. Note that the sum of the forcing vector elements represents the total heat flux through the body in this case. The various finite element models used are given in Table II.

TABLE II

FINITE ELEMENT MODELS FOR RADIAL CYLINDER PROBLEM

	Model			
	L-12	L-24	Q-6	C-4
Element Thickness (L,in)	1	1	1.5	1.5
Element Width (θ,°)	2	2	6	6
Element Type	Linear	Linear	Quadratic	Cubic
Number of Elements	12	24	6	4
Number of Nodes	52	100	80	92

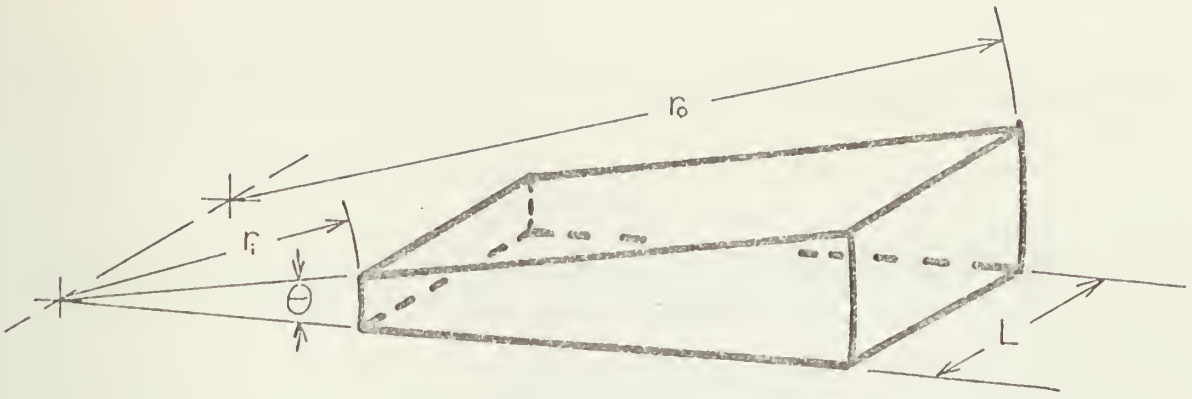


Figure 11. Radial Cylinder Problem Geometry.

TABLE III

RADIAL TEMPERATURE DISTRIBUTION IN A HOLLOW CYLINDER WITH THE INTERIOR SURFACE MAINTAINED AT 1500 °F

Radius (in)	Temperature (°F)				
	Theoretical	Model L-12	Model L-24	Model Q-6	Model C-4
3	1500.00	1500.00	1500.00	1500.00	1500.00
4	1188.54	1188.85	1188.63	1188.57	1188.53
5	946.96	947.34	947.06	947.01	946.99
6	749.56	749.94	749.67	749.63	749.62
7	582.67	583.00	582.77	582.75	582.74
8	438.11	438.38	438.20	438.19	438.19
9	310.59	310.81	310.67	310.69	310.68
Heat Flux (q)					
Theoretical		9.45	9.45	85.03	85.03
Calculated		9.46	9.46	85.03	85.02

TABLE IV

RADIAL TEMPERATURE DISTRIBUTION IN A HOLLOW CYLINDER WITH INTERNAL HEAT GENERATION

Radius (in)	Temperature (°F)			
	Theoretical	Model L-12	Model Q-6	Model C-4
3	175.28	175.34	175.30	175.30
4	173.00	173.03	173.02	173.02
5	166.77	166.79	166.79	166.79
6	157.13	157.13	157.15	157.14
7	144.35	144.34	144.36	144.36
8	128.60	128.60	128.62	128.62
9	110.00	109.99	110.02	110.02
Heat Flux (q)				
Theoretical		1.57	14.14	14.14
Calculated		1.57	14.14	14.14

3. Chimney Problem

Consider the two dimensional heat transfer problem posed by a rectangular chimney with the inside surface maintained at 100°F and the outside surface maintained at 0°F. The steady state temperature distribution may be obtained by constructing a finite element model consisting of the desired number and type of elements assembled to form a plate having the required geometry. The geometry of such a model utilizing symmetry boundary conditions is shown in Figure 12 and the results of various finite element models are compared with a solution obtained by a relaxation technique in Table V at the points shown in Figure 12.

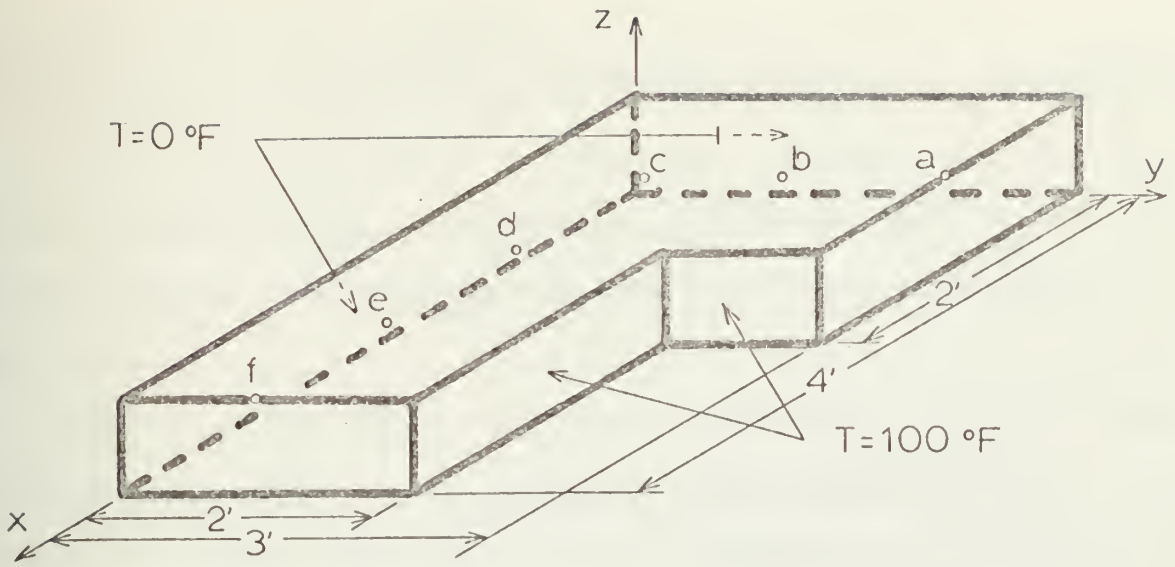


Figure 12. Rectangular Chimney Problem Geometry.

TABLE V

TEMPERATURE DISTRIBUTION IN A RECTANGULAR CHIMNEY

Point	(x,y)	Temperature (°F)			
		Relaxation Method	5 Cubic Element Model	10 Quadratic Element Model	40 Linear Element Model
a	1,3	45.5		44.3	45.2
b	1,2	38.9		37.5	38.3
c	1,1	20.6		19.4	20.3
d	2,1	39.2	37.1	37.9	38.7
e	3,1	47.7		47.0	47.5
f	4,1	49.3	48.7	48.9	49.1

C. FIRST ORDER TIME PROBLEMS

Transient heat conduction in various bodies is a common form of the field equation that is easily modeled by the finite element process. Transient heat conduction in a hollow cylinder and a prolate spheroid will be considered here.

Euler I, Trapezoidal, and Linear Time Shape Function time integration methods were employed. Each method yielded excellent results when used with an appropriate time step size. However, with the cylinder problem, the Euler I method was unstable for large time increments. The Trapezoidal and Linear Time Shape Function methods produced excellent results in all cases.

1. Transient Temperature Distribution in a Hollow Cylinder of Infinite Length

Consider the hollow cylinder of Steady State Test 2, Figure 11. Thermal diffusivity, α, is specified as .25 ft^2/hr where $\alpha = k/\rho c$. Initially the cylinder is at an ambient temperature of 70°F; suddenly the inside wall temperature is raised to 1500°F and maintained at this temperature. All other conditions are the same as for the Steady State Problem.

An approximate analytical solution, which is valid until the temperature wave propagates to the outer wall of the cylinder, can be found by applying methods of solution discussed in Ref. 1. The result is:

$$T = (T_o - T_\infty)\ \sqrt{r_i/r}\ \text{erfc}\left(\frac{r-r_i}{2\sqrt{\alpha t}}\right) + T_\infty$$

The finite element models employed here are identical to the models employed for the Steady State case. Transient solutions obtained by the Trapezoidal and Time Shape Function methods and using the linear element model converged to the same results obtained in the Steady State Problem for corresponding model. Solutions with Quadratic or Cubic models and Euler I integration were not pursued to steady state.

Tables VI through IX compare analytical and finite element solutions at various problem times for the 100 node, 24 Linear element model used with the Steady State Problem. Comparable results were obtained with Quadratic and Cubic element models. The following abbreviations apply to the time integration methods:

a. E-1 Euler I

b. TRAP Trapezoidal

c. LTS Linear Time Shape Function

TABLE VI

RADIAL TEMPERATURE DISTRIBUTION IN A CYLINDER AT .05 MINUTES

Method	Temperature (°F)						
	Exact	E-1	F-1	TRAP	LTS	TRAP	LTS
h(min)		.001	.0005	.001	.001	.01	.01
Radius(in)							
3.00	1500.00	1500.00	1500.00	1500.00	1500.00	1500.00	1500.00
3.25	492.38	595.41	591.70	593.06	592.26	636.45	629.17
3.50	124.58	153.61	151.81	153.45	153.40	183.64	182.41
3.75	72.81	63.84	63.99	64.64	64.91	69.94	72.42
4.00	70.06	68.16	68.28	70.42	70.40	67.36	68.01
5.00	70.00	70.01	70.01	70.00	70.00	70.00	70.00
6.00	70.00	70.00	70.00	70.00	70.00	70.00	70.00
9.00	70.00	70.00	70.00	70.00	70.00	70.00	70.00

TABLE VII

RADIAL TEMPERATURE DISTRIBUTION IN A CYLINDER AT .50 MINUTES

Method	Temperature (°F)						
	Exact	E-1	E-1	TRAP	LTS	TRAP	LTS
h(min)		.001	.0005	.001	.001	.01	.01
Radius(in)							
3.00	1500.00	1500.00	1500.00	1500.00	1500.00	1500.00	1500.00
3.25	1096.15	1105.20	1104.93	1104.98	1104.91	1107.83	1107.10
3.50	756.59	770.04	769.60	769.70	769.59	774.45	773.35
3.75	495.82	509.04	508.58	508.73	508.62	514.12	513.10
4.00	313.60	323.52	323.15	323.33	323.27	328.28	327.64
5.00	80.88	80.34	80.34	80.42	80.45	81.20	81.46
6.00	70.11	70.03	70.03	70.03	70.03	70.04	70.06
7.00	70.00	70.00	70.00	70.00	70.00	70.00	70.00
9.00	70.00	70.00	70.00	70.00	70.00	70.00	70.00

TABLE VIII

RADIAL TEMPERATURE DISTRIBUTION IN A CYLINDER AT 1.00 MINUTES

Method	Temperature (°F)						
	Exact	E-1	E-1	TRAP	LTS	TRAP	LTS
h(min)		.001	.0005	.001	.001	.01	.01
Radius(in)							
3.00	1500.00	1500.00	1500.00	1500.00	1500.00	1500.00	1500.00
3.25	1195.88	1201.16	1201.06	1201.08	1201.05	1202.15	1201.87
3.50	928.00	936.13	935.95	935.98	935.93	937.90	937.41
3.75	701.28	710.30	710.08	710.13	710.08	712.58	712.02
4.00	517.45	525.93	525.71	525.78	525.72	528.43	527.90
5.00	145.20	147.10	147.02	147.11	147.11	148.51	148.54
6.00	76.24	76.00	76.00	76.03	76.04	76.27	76.37
7.00	70.24	70.18	70.18	70.18	70.18	70.20	70.21
8.00	70.00	70.00	70.00	70.00	70.00	70.00	70.00
9.00	70.00	70.00	70.00	70.00	70.00	70.00	70.00

TABLE IX

RADIAL TEMPERATURE DISTRIBUTION IN A CYLINDER AT 1.50 MINUTES

Method	Temperature (°F)					
	Exact	E-1	TRAP	LTS	TRAP	LTS
h(min)		.0005*	.001**	.001**	.01	.01
Radius(in)						
3.00	1500.00	1500.00	1500.00	1500.00	1500.00	1500.00
3.25	1240.81	1245.47	1246.07	1246.00	1246.00	1245.84
3.50	1009.18	1016.34	1017.44	1017.33	1017.32	1017.04
3.75	806.91	815.06	816.53	816.38	816.37	816.01
4.00	634.79	642.85	644.53	644.38	644.35	643.98
5.00	220.69	224.17	225.54	225.50	225.39	225.29
6.00	95.63	95.97	96.47	96.50	96.42	96.50
7.00	72.69	72.57	72.66	72.68	72.65	72.70
8.00	70.17	70.14	70.15	70.15	70.15	70.15
9.00	70.01	70.01	70.01	70.01	70.01	70.01

* .0005 to 1 min, .001 thereafter

** .001 to 1 min., .01 thereafter

2. Transient Heat Conduction in a Prolate Spheroidal Solid

An analytical solution for transient heat conduction in a prolate spheroidal solid is developed in Ref. 4. Results of a finite element solution to this problem are presented in Refs. 10 and 11.

A non-dimensional finite element model using one cubic element or four Quadratic elements was constructed. The Linear Time Shape Function of time integration was employed giving results comparable to the results of Refs. 10 and 11. A comparison of theoretical and finite element results is presented in Figure 13.

D. SECOND ORDER TIME PROBLEMS

The transient vibration of a perfectly elastic membrane is a two-dimensional problem that can be formulated as a finite element model of the field equation.

Consider the square elastic membrane of length a and specified material properties T_o, ρ, and ν. T_o is the initial tension of the membrane, ρ is the surface density, ν is the damping factor, and $c^2 = T_o/\rho$. The partial differential equation describing the motion is:

$$c^2 \frac{\partial^2 u}{\partial x^2} + c^2 \frac{\partial^2 u}{\partial y^2} - 2\nu \frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial t^2} = 0$$

where $u(x,y,t)$ is the displacement of the membrane.

An exact mathematical solution is derived in Ref. 2 and may be specialized to a simple closed form if an initial displacement of the form

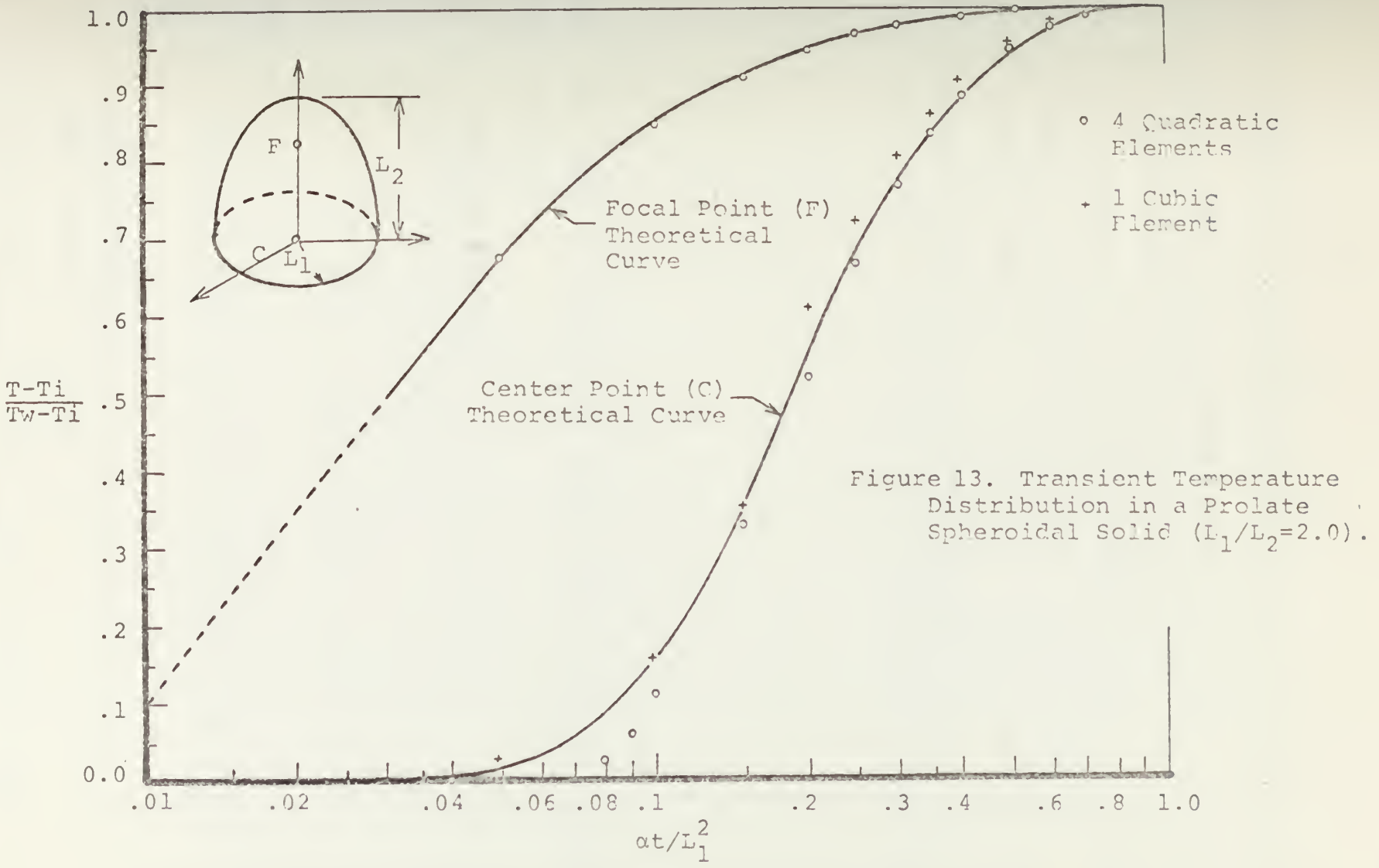


Figure 13. Transient Temperature Distribution in a Prolate Spheroidal Solid (L_1/L_2=2.0).

$$u(x,y,0) = d \sin \frac{\pi x}{a} \sin \frac{\pi y}{a}$$

and boundary conditions of simply supported edges are assumed.

Two finite element models were constructed. One models the entire membrane as a 36 Linear element plate; the other as a 9 Quadratic element quarter plate imposing symmetry boundary conditions at the centerlines. Euler I and Finite Difference solution techniques were employed with a time step size of .001. Assume the following material properties:

a. $a = 1.5$

b. $c^2 = 4.5$

c. $\nu = 3.0$

d. $d = .25$

Both time integration methods produced good results; however, the Euler I integration method was unstable at large time increments.

1. Undamped Membrane

The analytical solution is

$$\mu = d \cos \frac{2\pi ct}{a} \sin \frac{\pi x}{a} \sin \frac{\pi y}{a}$$

Results obtained with the finite element models are compared to the analytical result for the center node displacement in Table X.

2. Damped Membrane

The analytical solution is

$$\mu = d\, e^{-\nu t} \left(\cos \omega t + \frac{\nu}{\omega} \sin \omega t\right) \sin \frac{\pi x}{a} \cos \frac{\pi y}{a}$$

TABLE X

CENTER NODE DISPLACEMENT OF AN UNDAMPED MEMBRANE

	Displacement				
		EULER I Method		FINITE DIFFERENCE Method	
Time	Exact	Linear Model	Quadratic Model	Linear Model	Quadratic Model
.00	.2500	.2500	.2500	.2500	.2500
.05	.2378	.2372	.2370	.2370	.2368
.10	.2023	.2007	.2108	.2002	.2014
.15	.1469	.1439	.1456	.1435	.1453
.20	.0773	.0726	.0763	.0723	.0759
.25	.0000	-.0061	-.0020	-.0061	-.0018
.30	-.0773	-.0844	-.0787	-.0839	-.0784
.35	-.1469	-.1543	-.1496	-.1532	-.1483
.40	-.2023	-.2090	-.2041	-.2073	-.2027
.45	-.2378	-.2428	-.2407	-.2406	-.2383
.50	-.2500	-.2523	-.2516	-.2498	-.2495

where

$$\omega = \sqrt{\lambda^2 c^2 - \nu^2} \quad \text{and} \quad \lambda^2 = \pi/a$$

Results obtained with the finite element models are compared to the analytical result for the center node displacement in Table XI.

TABLE XI

CENTER NODE DISPLACEMENT OF A DAMPED MEMBRANE

		Displacement			
		EULER I Method		FINITE DIFFERENCE Method	
Time	Exact	Linear Model	Quadratic Model	Linear Model	Quadratic Model
.00	.2500	.2500	.2500	.2500	.2500
.05	.2389	.2384	.2382	.2382	.2380
.10	.2105	.2091	.2100	.2089	.2097
.15	.1716	.1692	.1705	.1691	.1704
.20	.1283	.1249	.1275	.1250	.1274
.25	.0854	.0813	.0841	.0815	.0843
.30	.0464	.0418	.0454	.0422	.0457
.35	.0135	.0089	.0122	.0095	.0128
.40	-.0119	-.0162	-.0128	-.0156	-.0123
.45	-.0298	-.0335	-.0309	-.0328	-.0301
.50	-.0405	-.0434	-.0411	-.0427*	-.0406*
.55	-.0450		**	-.0588	-.0587
.60	-.0445			-.0639	-.0651
.65	-.0404			-.0621	-.0644
.70	-.0340			-.0555	-.0585
.75	-.0263			-.0457	-.0491

*Changed step size to .01

**Changed step size to .01 at time = .5, Solution became unstable

V. CONCLUSIONS AND RECOMMENDATIONS

The computer program presented in this thesis provides an accurate and reliable means for solving a variety of problems. The use of this program and efforts to increase its versatility are highly encouraged.

The Linear Time Shape Function and Four Point Backward Finite Difference (Houbolt) time integration methods provide the most consistent and reliable results. Investigation of other time integration schemes should be pursued.

A. RECOMMENDED MODIFICATIONS TO THE PROGRAM

A number of modifications to the present program are desirable in order to increase the versatility of the program. Most of these modifications are straightforward and are the result of the simplifying assumptions made during development of the program. The recommended modifications should present no major programming problems.

1. Anisotropic Materials

A facility for coordinate axis rotation will provide the program with the ability to accommodate problems in which principal material axes do not align. This may be accomplished by the addition of an appropriate subroutine to perform the axis rotation. This subroutine should be called from MERGE at the time of element formation. Minor changes to subroutine INPUT will be required.

2. Time Dependent Boundary Conditions

The ability to account for a time variation in the boundary value specified for $\{\phi\}$ is highly desirable. In principle the modifications required are straightforward. The required modifications will only affect the solution subroutines TIME11, TIME13, TIME21, and TIME23 if data input is accomplished by providing updated values of $\{\phi\}$ and $\{\dot{\phi}\}$, if required, in tabular form. The modifications will require attention to detail and a thorough understanding of the Irons technique as applied in the program. At present, boundary values of $\{\dot{\phi}\}$ are considered to be zero; this effect on the solution matrix corrections and the effect of time dependence of the {F} vector must be carefully considered.

Time dependence of material properties implies the recalculation of problem matrices at each station in time and possible iteration. Considerable computational effort is required and efficient programming techniques are necessary.

3. Non Linear Material Properties and Boundary Conditions

A functional dependence of material properties and non linear boundary conditions such as the radiation boundary condition of the heat problem are real physical situations. The ability of the program to accommodate this situation is needed and feasible by various iteration schemes.

4. Problem Size

The program is presently designed for problems of up to 110 equations with a maximum band width of 64. Practical

physical problems will require much more computer storage space. Modifications to accommodate more equations in core require a careful adjustment of the appropriate common and dimension statements and attention to the equivalence statements in the time integration subroutines.

5. Specialized Data Input and Output

Hand preparation of input data is tedious and subject to numerous errors. For practical physical problems of even moderate size, automated data input is a necessity.

Each class of physical problem will require data output in a specific form. Considerable attention should be given to the development of various subroutines to tailor data output to a given class of problem.

B. RECOMMENDED FUTURE STUDIES

This thesis was concerned with the development of a computer program with the capability of solving a class of physical problems. This objective was accomplished and recommendations for improvements stated. Questions raised during the course of this development and beyond the scope of this work are:

1. Convergence criteria of the various time integration methods, particularly the Euler I technique.

2. Selection criteria for finite element type and, in particular, the type of element best suited to a given geometry or physical situation.

3. Adaptability and usefulness of other finite element types.

APPENDIX A

COMPUTER PROGRAM NOMENCLATURE

A complete listing and description of all variables used in the program is not practical. The items listed in this appendix are common to several areas of the program and will assist the reader in a study of the program. For convenience items are listed by variable type. The definition or function and dimension, if applicable, of each item is given.

A. INTEGER CONSTANTS

IGO	Element type
IPUN	Punched output request
IPROB	Problem type
IPRT	Number of time increment steps to skip between print out of results
KBS	Boundary condition surface
KBT	Boundary condition type
KC	Coordinate system of reference
KINT	Time integration type
MRH	Time step size index
NBAND	System band width
NEL	Number of elements
NELBC	Number of element boundary conditions imposed
NGP	Number of Gauss points
NMAT	Number of materials
NNBC	Number of nodal boundary values specified

NNODE	Number of nodes
NPEL	Number of nodes per element
NTSC	Number of time step size increments specified
NTP	Number of times $\{\phi\}$ is to be printed

B. REAL CONSTANTS

ABIGN	Arbitrarily large number
AKX	[H] matrix property in the x direction
AKY	[H] matrix property in the y direction
AKZ	[H] matrix property in the z direction
APZRO	Calculated value of zero
DT	Time step size
DTJ	Determinant of the Jacobian
FACT	[C] or [G] matrix material property
PHIREF	Reference value of ϕ
TIM	Initial problem time
WFACT	Products of Gauss weight factors
WX,WY,WZ	Gauss weight factors for specified direction
XI,ETA,ZETA	Local coordinates ξ, η, and ζ
X1,Y1,Z1 or XX,YY,ZZ	Global coordinates x,y, and z

C. VECTORS

COL(32)	Element level storage of the shape function
ENDT(5)	End time of the various time integration steps
DTIM(5)	Time increment for each time step size
FE(32)	Forcing vector at element level
F(110)	Problem forcing vector

IP(5) IPRT for each time integration step size

KBCOND(48) KBT for each element

KEL(48) Element number for application of boundary conditions

KNODE(110) Node number for specified $\{\phi_B\}$

KSURF(48) KBS for each element

PHI(110) $\{\phi\}$ vector

PHIDOT(110) $\{\dot{\phi}\}$ vector

PHIDDT(110) $\{\ddot{\phi}\}$ vector

PHIWRK(110) Working vector

TIME(6) Time label for output

D. MATRICES

AJ(3,3) [J] matrix

AJI(3,3) $[J]^{-1}$ matrix

BGC(110,64) [C] matrix

BGG(110,64) [G] matrix

BGM(110,64) [C] or [H] matrix, depending on problem type

COARD(110,3) Global coordinates

CORD(32,3) Global coordinates for element calculations

DCOL(32,3) Derivatives of the shape functnon with respect to local coordinates. Column I is $\partial/\partial\xi$; Column II is $\partial/\partial\eta$; Column III is $\partial/\partial\zeta$.

ENN(32,32) Work matrix for accumulation of $[H]^e$, $[C]^e$, or $[G]^e$.

ENW(32,32) Work matrix for element level calculations

NCON(48,33) Connectivity matrix for each element, column 1 is material number

PROP(10,5) Material properties

OUT(110,6) Equivalenced matrix for storage of output data

GP(6,5)	Gauss points
WF(6,5)	Gauss weighting factors

APPENDIX B

INSTRUCTIONS FOR PROBLEM DECK PREPARATION

Although the procedure is straightforward, preparation of input data for the program requires meticulous attention to detail. Errors are easy to make and difficult to locate.

Input data preparation should, in general, follow the steps outlined below before attempting to punch data cards. The use of the standard Fortran Eighty Column Coding Sheet is highly recommended. A sample problem deck follows this appendix. Integer constants must be right adjusted in the appropriate field.

Preliminary preparation:

1. Formulate the problem in terms of equation (1) and boundary conditions in terms of equations (2) and (3). Identify the appropriate material properties. In the case of a one or two-dimensional problem, certain properties must be specified as 1.0; a specification of 0.0 will lead to a singular system of equations.

2. Divide the continuum into the desired number and type of elements. Observe the restrictions of 48 elements, 110 nodes, 64 bandwidth, 10 materials, and 48 element level boundary conditions.

3. Assign node numbers as desired, observing the band width restriction.

4. List the element connectivity in accordance with the conventions of Figures 1, 2, or 3. View ξ, η, and ζ as x, y, and z respectively. Any other convention will lead to serious error.

5. List all nodal points and coordinates in any convenient order.

6. List the applicable boundary conditions and the appropriate codes and material properties.

7. If applicable, select the solution subroutine desired and list the required time integration parameters. For the Houbolt method the specification of ENDT must allow enough time to generate the starting values required for the next step in time.

Preliminary preparations are now complete; precise card punching instructions follow. Card format is given in parenthesis followed by specific instructions where necessary. Recall that blank columns are read as zero.

1. Title Card (10A8)

Eighty columns on one card for any desired title information. Column 1 of this card will not be printed; if a 1 is punched, the output starts on a new page.

2. Problem Parameters (10I5,10X,F20.0)

One card containing the following data:

a. Cols 1-5 : Number of nodes (NNODE)

b. Cols 6-10 : Number of elements (NEL)

c. Cols 11-15 : Number of materials (NMAT)

d. Cols 16-20 : Number of Gauss points (NGP)
Specify 2 through 6; default to 3

e. Cols 21-25 : Element type

1 = Linear element

2 = Quadratic element

3 = Cubic element

f. Cols 26-30 : Number of element level boundary conditions (NELBC)

g. Cols 31-35 : Number of nodal boundary values specified (NNBC)

h. Cols 36-40 : Problem type (IPROB)

1 = Steady state

2 = First order in time

3 = Second order in time

4 = Second order in time without [C] matrix

i. Cols 41-45: Coordinate system of reference (KC)

1 = Rectangular coordinates

2 = Cylindrical coordinates

3 = Spherical coordinates

j. Cols 46-50: Punched output requested (IPUN)

0 = No punched output desired

1 = Punched output desired

k. Cols 61-80: Reference value of ϕ (PHIREF)

3. Material Properties; One card per material for the steady state problem or two cards per material for the transient problem.

a. Card I (I10,3F20.0)

Cols 1-10: Material number

Cols 11-30: k_x

Cols 31-50: k_y

Cols 51-70: k_z

b. Card II (I10,2F20.0)

Cols 1-10: Material number

Cols 11-30: [G] matrix coefficient

Cols 31-50: [C] matrix coefficient

4. Node cards; One set of cards for the steady state problem or two sets of cards for the transient problem. Each set has NNODE cards.

a. Set I (I10,3F20.0)

Cols 1-10: Node number

Cols 11-30: x coordinate or radius

Cols 31-50: y coordinate or angle in degrees relative to the x axis (+CCW)

Cols 51-70: z coordinate or angle in degrees relative to the z axis (+CW)

b. Set II (I10,2F20.0)

Cols 1-10: Node number

Cols 11-30: Initial value of ϕ_i

Cols 31-50: Initial value of $\dot{\phi}_i$

Leave blank if first order time problem.

5. Node boundary values; NNBC cards required but omit if NNBC = 0 (I10,F20.0)

Cols 1-10: Node number

Cols 11-30: Specified value of ϕ_i

6. Connectivity data; NEL cards required for the Linear or Quadratic element and NEL x 2 cards required for the Cubic element (2I5,32I3)

a. Card I(2I5,23I3)

Cols 1-5: Element number

Cols 6-10: Material identification number

Cols 11-79: Connectivity for nodes 1 through 23

b. Card II (9I3): Use with Cubic element only

Cols 1-27: Connectivity for cubic element nodes 24-32

7. Boundary condition data; NELBC cards required but omit if NELBC = 0 (3I5,F20.0)

Cols 1-5: Element number

Cols 6-10: Boundary condition type (KBT)

1 = Volume integral of loading (Q)

2 = Surface integral of loading (q)

3 = Surface loss coefficient (α)

Cols 11-15: Type of integral and surface applicable (KBS)

0 = Volume integral; Use with KBT=1 only

$\pm$1 = Surface integral over $\xi=\pm$ 1 face

$\pm$2 = Surface integral over $\eta=\pm$ 1 face

$\pm$3 = Surface integral over $\zeta=\pm$ 1 face

Note: KBS = 1,2, or 3 must be signed plus (+) or minus (-) to indicate the applicable integration surface

Cols 16-35: Boundary condition property value

8. Transient problem data; NTSC + 1 cards required but omit for the steady state problem.

a. Card I, one card (2I10,10X,F20.0)

Cols 1-10: Number of changes in time step size (NTSC)

Cols 11-20: Time integration method (KINT) Default to method 3

1 = Euler I method

2 = Trapezoidal rule method

3 = Linear Time Shape Function or
Backward Finite Difference method

Cols 31-50: Initial problem time

b. Set II, NTSC cards (I10,2F20.0)

Cols 1-10: Number of time intervals to skip before printing output (IPRT)

Cols 11-30: Time interval (DT)

Cols 31-50: Total problem time to end of interval (ENDT)

Data preparation is now complete. Cards should be arranged in order of ascending node numbers within each set or group and each set or group of cards stacked in the order prepared. Refer to the sample problem deck following this appendix.

Several problems may be stacked and submitted if desired. In order to terminate a given computer run add one last data card with the word "STOP" punched in columns 1-4. Standard JCL cards and deck structure is shown in Appendix C. Print out of the program is suppressed by the "//FORT. SYSPRINT DD DUMMY" card; if a program listing is desired remove this card. The JCL card requesting additional output should be removed unless voluminous output is expected. Good Luck!

SAMPLE PROBLEM DECK

```
        1         2         3         4         5         6         7         8
12345678901234567890123456789012345678901234567890123456789012345678901234567890

1 STEADY STATE AXIAL HEAT FLOW IN AN INSULATED CYLINDER (FLUX/TOP) <BTU/HR/FT>
   32     2    1    4    2    1    8    1    1    0          300.
          1 20.                  20.                  20.
          1 2.82842712474619     2.82842712474619     4.
          2 1.52814988653234     3.696587334865926    4.
          3 0.                   4.                   4.
          4 0.                   2.                   4.
          5 0.                   0.                   4.
          6 2.                   0.                   4.
          7 4.                   0.                   4.
          8 3.696587334865926    1.52814988653234     4.
          9 2.82842712474619     2.82842712474619     3.
         10 0.                   4.                   3.
         11 0.                   0.                   3.
         12 4.                   0.                   3.
         13 2.82842712474619     2.82842712474619     2.
         14 1.52814988653234     3.696587334865926    2.
         15 0.                   4.                   2.
         16 0.                   2.                   2.
         17 0.                   0.                   2.
         18 2.                   0.                   2.
         19 4.                   0.                   2.
         20 3.696587334865926    1.52814988653234     2.
         21 2.82842712474619     2.82842712474619     1.
         22 0.                   4.                   1.
         23 0.                   0.                   1.
         24 4.                   0.                   1.
         25 2.82842712474619     2.82842712474619     0.
         26 1.52814988653234     3.696587334865926    0.
         27 0.                   4.                   0.
         28 0.                   2.                   0.
         29 0.                   0.                   0.
         30 2.                   0.                   0.
         31 4.                   0.                   0.
         32 3.696587334865926    1.52814988653234     0.
```

```
        1         2         3         4         5         6         7         8
12345678901234567890123456789012345678901234567890123456789012345678901234567890

          25 -200.
          26 -200.
          27 -200.
          28 -200.
          29 -200.
          30 -200.
          31 -200.
          32 -200.
     1     1  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
     2     1 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
     1     2     3 -500.
STOP
```

APPENDIX C

STANDARD DECK STRUCTURE

```
//LEWSTHSI JOB (2032,0773FP,NS12),'LEW,GIRARD T.',TIME=(1,30)
// EXEC FORTHCLG,REGION.GO=285K
//FORT.SYSPRINT DD DUMMY
//FORT.SYSIN DD *

 Insert program deck at this point.  Omit "DUMMY" card if
a program listing is desired.  Omit "SPACE" card if a large
number of outout pages is not required

/*
//GO.FT06F001 DD SPACE=(CYL,(4,1))
//GO.FT10F001 DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(NEW,DELETE)
//GO.FT11F001 DD UNIT=SYSDA,SPACE=(CYL,(1,1)),DISP=(NEW,DELETE)
//GO.SYSIN DD *

Insert data deck here

/*
```

APPENDIX D

COMPUTER LISTING

```
C     ************************************************************** MAN0010
C                                                                      MAN0020
C     PROGRAM "FIELD" , A SOLUTION TO THE TRANSIENT FIELD EQUATION USING MAN0030
C                       ISOPARAMETRIC FINITE ELEMENTS                  MAN0040
C        RESTRICTIONS                                                  MAN0050
C             1:  TIME DEPENDENCE IN BOUNDARY CONTITIONS NOT ALLOWED   MAN0060
C             2:  STATIONARY ELEMENTS AND BOUNDARIES ARE REQUIRED      MAN0070
C             3:  IF ANISOTROPIC THE PRINCIPAL AXES MUST ALIGN         MAN0080
C                                                                      MAN0090
C     INPUT DATA REQUIRED                                              MAN0100
C                                                                      MAN0110
C        (1) ONE TITLE CARD (10A8)                                     MAN0120
C                                                                      MAN0130
C        (2) PROBLEM DATA, ONE CARD (10I5,10X,F20.0)                   MAN0140
C             1: NUMBER OF NODES (MAX IS 110) <NNODE>                  MAN0150
C                MAX BAND WIDTH = 64  <NBAND>                          MAN0160
C             2: NUMBER OF ELEMENTS (MAX IS 48) <NEL>                  MAN0170
C             3: NUMBER OF MATERIALS (MAX IS 10) <NMAT>                MAN0180
C             4: NUMBER OF GAUSS POINTS (2 TO 6; DEFAULT TO 3) <NGP>   MAN0190
C             5: CODE TO INDICATE ELEMENT TYPE:                        MAN0200
C                  1 = LINEAR ELEMENT                                  MAN0210
C                  2 = QUADRATIC ELEMENT                               MAN0220
C                  3 = CUBIC ELEMENT                                   MAN0230
C             6: NUMBER OF ELEMENT BOUNDARY CONDITIONS (48 MAX) <NELBC>MAN0240
C             7: NUMBER OF NODES WITH BOUNDARY VALUES SPECIFIED <NNBC> MAN0250
C             8: PROBLEM TYPE <IPROB>                                  MAN0260
C                  1 = STEADY STATE                                    MAN0270
C                  2 = FIRST ORDER IN TIME                             MAN0280
C                  3 = SECOND ORDER IN TIME                            MAN0290
C                  4 = SECOND ORDER IN TIME, NO FIRST ORDER DERIVATIVE MAN0300
C             9: INPUT SYSTEM OF REFERENCE <KC>                        MAN0310
C                  1 = RECTANGULAR COORDINATES                         MAN0320
C                  2 = CYLINDRICAL COORDINATES                         MAN0330
C                  3 = SPHERICAL   COORDINATES                         MAN0340
C            10: PUNCHED OUTPUT REQUESTED <IPUN>                       MAN0350
C                A POSITIVE INTEGER WILL PRODUCE AN OUTPUT OF NNODE    MAN0360
C                CARDS IN FORMAT (I10,2D24.16).  NODE NUMBER IS        MAN0370
C                FOLLOWED BY THE VALUE OF "PHI" AND "PHIDOT" (IF       MAN0380
C                APPLICABLE).  OUTPUT IS PRODUCED AT THE END OF EACH   MAN0390
C                TIME INTERVAL CHANGE FOR THE TRANSIENT PROBLEM.       MAN0400
C            11: REFERENCE VALUE OF PHI <PHIREF> (COLS 61-80)          MAN0410
C                SET AT ZERO IF NOT SPECIFIED                          MAN0420
C                                                                      MAN0430
```

```
C          (3) MATERIAL PROPERTIES; NMAT X TWO CARDS                          MAN0440
C              (DATA ASSUMED ZERO IF NOT SPECIFIED)                           MAN0450
C                                                                             MAN0460
C              CARD  I:                                                       MAN0470
C               COLS  1-10: MATERIAL NUMBER                                   MAN0480
C               COLS 11-30: KX                                                MAN0490
C               COLS 31-50: KY                                                MAN0500
C               COLS 51-70: KZ                                                MAN0510
C                                                                             MAN0520
C              CARD II: (NOT REQUIRED FOR THE STEADY STATE PROBLEM)           MAN0530
C               COLS  1-10: MATERIAL NUMBER                                   MAN0540
C               COLS 11-30: DENSITY ("G" <SECOND TIME DERIATIVE> MATRIX       MAN0550
C                           COEFFICIENT)                                      MAN0560
C               COLS 31-50: VISCOSITY ("C" <FIRST TIME DERIVATIVE> MATRIX     MAN0570
C                           COEFFICIENT)                                      MAN0580
C                                                                             MAN0590
C          (4) NODE CARDS; TWO SETS OF NNODE CARDS PER PROBLEM                MAN0600
C              (DATA ASSUMED ZERO IF NOT SPECIFIED)                           MAN0610
C                                                                             MAN0620
C              SET  I:                                                        MAN0630
C               COLS  1-10 NODE NUMBER                                        MAN0640
C               COLS 11-30 X COORDINATE OR RADIUS                             MAN0650
C               COLS 31-50 Y COORDINATE OR ANGLE IN DEGREES RELATIVE TO       MAN0660
C                          X AXIS (+ CCW)                                     MAN0670
C               COLS 51-70 Z COORDINATE OR ANGLE IN DEGREES RELATIVE TO       MAN0680
C                          Z AXIS (+ CW)                                      MAN0690
C                                                                             MAN0700
C              SET II: (NOT REQUIRED FOR THE STEADY STATE PROBLEM)            MAN0710
C               COLS  1-10: NODE NUMBER                                       MAN0720
C               COLS 11-30: INITIAL VALUE OF PHI                              MAN0730
C               COLS 31-50: INITIAL VALUE OF PHIDOT                           MAN0740
C                                                                             MAN0750
C          (5) NODE BOUNDARY VALUES; NNBC CARDS PER PROBLEM                   MAN0760
C              INPUT IN ORDER OF ACENDING NODE NUMBERS                        MAN0770
C              (OMIT IF DATA NOT REQUIRED)                                    MAN0780
C               COLS  1-10  JOINT NUMBER                                      MAN0790
C               COLS 11-30  SPECIFIED NODAL VALUE                             MAN0800
C                                                                             MAN0810
C          (6) CONNECTIVITY DATA, NEL CARDS PER PROBLEM (2I5,32I3)            MAN0820
C               COLS  1-5  ELEMENT NUMBER                                     MAN0830
C               COLS  6-10 MATERIAL IDENTIFICATION NUMBER                     MAN0840
C               COLS 11-107 CONNECTIVITY, CCW FROM UPPER RIGHT                MAN0850
C                           LOOKING INTO Y-Z PLANE FROM (+) X AXIS            MAN0860
C                           NODES  1-23 ON CARD I (2I5,23I3)                  MAN0870
C                           NODES 24-32 ON CARD II (9I3)                      MAN0880
C                                                                             MAN0890
C          (7) BOUNDARY CONDITION DATA; NELBC CARDS PER PROBLEM (3I5,F20.0)   MAN0900
C              (OMIT IF DATA NOT REQUIRED)                                    MAN0910
```

```
C                1:  ELEMENT NUMBER (COLS 1-5)                                  MAN0920
C                2:  CODE TO INDICATE BOUNDARY CONDITION TYPE (KBT)             MAN0930
C                      1 = VOLUME INTEGRAL OF LOADING                           MAN0940
C                      2 = SURFACE INTEGRAL OF LOADING                          MAN0950
C                      3 = SURFACE LOSS COEFFICIENT                             MAN0960
C                3:  CODE TO INDICATE TYPE OF INTEGRAL REQUIRED AND             MAN0970
C                    SURFACE APPLICABLE (KBS)                                   MAN0980
C                      0 = VOLUME INTEGRAL (USE WITH KBT = 1 ONLY)              MAN0990
C                      1 = SURFACE INTEGRAL OVER XI = +/- 1 FACE                MAN1000
C                      2 = SURFACE INTEGRAL OVER ETA = +/- 1 FACE               MAN1010
C                      3 = SURFACE INTEGRAL OVER ZETA = +/- 1 FACE              MAN1020
C                      *** KBS = 1,2 OR 3 MUST BE SIGNED TO INDICATE ***        MAN1030
C                      *** PLUS (+) OR MINUS (-) FACE                ***        MAN1040
C                4:  BOUNDARY CONDITION PROPERTY VALUE (COLS 16-35)             MAN1050
C                                                                               MAN1060
C          (8) TRANSIENT PROBLEM DATA, NSTC + 1 CARDS                           MAN1070
C              (OMIT IF NOT REQUIRED)                                           MAN1080
C                                                                               MAN1090
C              CARD I: (2I10,10X,F20.0)                                         MAN1100
C               COLS  1-10: NUMBER OF CHANGES IN TIME STEP SIZE <NTSC>          MAN1110
C                           (MAXIMUM OF FIVE INCREMENTS ALLOWED)                MAN1120
C               COLS 11-20: TIME INTEGRATION METHOD <KINT>                      MAN1130
C                           (DEFAULT TO METHOD THREE)                           MAN1140
C                     1 = EULER I (FIRST AND SECOND ORDER TIME PROBLEM)         MAN1150
C                     2 = TRAPEZOIDAL (FIRST ORDER TIME PROBLEM ONLY)           MAN1160
C                     3 = TIME SHAPE FUNCTION (FIRST ORDER TIME PROBLEM)        MAN1170
C                     3 = HOUBOLT (SECOND ORDER TIME PROBLEM)                   MAN1180
C                           (FOR HOUBOLT INTEGRATION METHOD "ENDT" MUST         MAN1190
C                           ALLOW SUFFICIENT TIME TO GENERATE THE               MAN1200
C                           STARTING VALUES FOR THE NEXT TIME STEP              MAN1210
C                           SIZE CHANGE)                                        MAN1220
C               COLS 31-50: INITIAL PROBLEM TIME                                MAN1230
C                           (SET AT ZERO IF NOT SPECIFIED)                      MAN1240
C                                                                               MAN1250
C              SET II: NTSC CARDS PER PROBLEM (I10,2F20.0)                      MAN1260
C               COLS  1-10: NUMBER OF TIME INTERVALS TO SKIP BEFORE             MAN1270
C                           PRINTING <IPRT>                                     MAN1280
C               COLS 11-30: TIME INTERVAL                                       MAN1290
C               COLS 31-50: TOTAL PROBLEM TIME TO END OF INTERVAL               MAN1300
C                                                                               MAN1310
C     STACK PROBLEMS IF DESIRED.  TO TERMINATE RUN ADD DATA CARD                MAN1320
C     WITH "STOP" PUNCHED IN COLS 1-4                                           MAN1330
C                                                                               MAN1340
C*****CODED IN NOVEMBER 1972 BY GIRARD T. LEW, LCDR, USN**********************  MAN1350
C                                                                               MAN1360
C     ***************************************************************           MAN1370
      IMPLICIT REAL*8(A-H,O-Z)                                                  MAN1380
      COMMON /MTRX/ CORD(32,3),ENN(32,32),DCOL(32,3),COL(32),AJ(3,3),AJI        MAN1390
```

```
     1(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COARD(110 MAN1400
     2,3),F(110),PHI(110),PHIDOT(110),PHIWRK(110),PHIDDT(110),DTIM(5),EN MAN1410
     3DT(5),BGH(110,64),BGC(110,64)                                      MAN1420
      COMMON /FLP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM           MAN1430
      COMMON /INT/ IGO,NBAND,NNODE,NMAT,NEL,NPEL,NGP,NELBC,NNBC,IPROB,KC MAN1440
     1,NTSC,KINT,IPJN                                                    MAN1450
      COMMON /IMTRX/ NCON(48,33),KEL(48),KBCOND(48),KSURF(48),KNODE(110) MAN1460
     1,IP(5)                                                             MAN1470
  100 CALL INPUT                                                         MAN1480
      GO TO (200,300,400,400),IPROB                                      MAN1490
C                                                                        MAN1500
C     STEADY STATE PROBLEM                                               MAN1510
C                                                                        MAN1520
  200 CALL MERGE(1)                                                      MAN1530
      CALL STEADY                                                        MAN1540
      GO TO 100                                                          MAN1550
C                                                                        MAN1560
C     FIRST ORDER TIME DERIVATIVE PROBLEM                                MAN1570
C                                                                        MAN1580
  300 CALL MERGE(1)                                                      MAN1590
      CALL MERGE(2)                                                      MAN1600
      IF(KINT.EQ.1) CALL TIME11                                          MAN1610
      IF(KINT.EQ.2.OR.KINT.EQ.3) CALL TIME13                             MAN1620
      GO TO 100                                                          MAN1630
C                                                                        MAN1640
C     SECOND ORDER TIME DERIVATIVE PROBLEM                               MAN1650
C                                                                        MAN1660
  400 CALL MERGE(1)                                                      MAN1670
      IF(IPROB.EQ.3) CALL MERGE(2)                                       MAN1680
      CALL MERGE(3)                                                      MAN1690
      IF(KINT.EQ.1) CALL TIME21                                          MAN1700
      IF(KINT.EQ.2.OR.KINT.EQ.3) CALL TIME23                             MAN1710
      GO TO 100                                                          MAN1720
      END                                                                MAN1730
```

```
      SUBROUTINE INPUT                                                      IPT0010
      IMPLICIT REAL*8(A-H,O-Z)                                              IPT0020
      COMMON /MTRX/ CORD(32,3),ENN(32,32),DCOL(32,3),COL(32),AJ(3,3),AJI    IPT0030
     1(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COARD(110    IPT0040
     2,3),F(110),PHI(110),PHIDOT(110),PHIWRK(110),PHIDDT(110),DTIM(5),EN    IPT0050
     3DT(5),BGH(110,64),BGC(110,64)                                         IPT0060
      COMMON /FLP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM              IPT0070
      COMMON /INT/ IGO,NBAND,NNODE,NMAT,NEL,NPEL,NGP,NELBC,NNBC,IPROB,KC    IPT0080
     1,NTSC,KINT,IPUN                                                       IPT0090
      COMMON /IMTRX/ NCON(48,33),KEL(48),KBCOND(48),KSURF(48),KNODE(110)    IPT0100
     1,IP(5)                                                                IPT0110
      DIMENSION TITLE (10)                                                  IPT0120
      DATA CHK/'STOP    '/                                                  IPT0130
      READ (5,1000) TITLE                                                   IPT0140
 1000 FORMAT (10A8)                                                         IPT0150
      IF (TITLE(1).EQ.CHK) STOP                                             IPT0160
C
C     READ THE BASIC PROBLEM DATA (10I5,10X,F20.0)
C
      WRITE (6,1000) TITLE                                                  IPT0170
      READ(5,1100) NNODE,NEL,NMAT,NGP,IGO,NELBC,NNBC,IPROB,KC,IPUN,PHIRE    IPT0180
     1F                                                                     IPT0190
 1100 FORMAT(10I5,10X,F20.0)                                                IPT0200
      WRITE (6,1200) NNODE,NEL,NMAT,NGP,NELBC,NNBC,IPROB,PHIREF             IPT0210
 1200 FORMAT(///,11X,' NUMBER OF NODES = ',I5,//,11X,' NUMBER OF ELEMENT    IPT0220
     1S = ',I5,//,11X,' NUMBER OF MATERIALS = ',I5,//,11X,' NUMBER OF GA    IPT0230
     2USS POINTS = ',I5,//,11X,' NUMBER OF ELEMENT BOUNDARY CONDITIONS =    IPT0240
     3 ',I5,//,11X,' NUMBER OF NODES WITH BOUNDARY VALUE = ',I5,//,11X,'    IPT0250
     4 PROBLEM TYPE = ',I5,//,11X,' PHI(REFERENCE) = ',1PG24.16)            IPT0260
C
C     COMPUTE THE NUMBER OF NODES PER ELEMENT FROM ELEMENT TYPE
C
      NPEL=12*(IGO-1)+8                                                     IPT0270
      WRITE (6,1300) IGO,NPEL                                               IPT0280
 1300 FORMAT (/,11X,' TYPE',I3,'  ELEMENTS,',I5,'   POINTS PER ELEMENT')    IPT0290
      IF(IPUN.GE.1) WRITE(6,1310)                                           IPT0300
 1310 FORMAT(//,11X,' PUNCHED OUTPUT OF "PHI" AND "PHIDOT" REQUESTED')      IPT0310
C
C     ZERO THE DATA STORAGE MATRICES
C
      NP1=NPEL+1                                                            IPT0320
      DO 1320 I=1,NEL                                                       IPT0330
      DO 1320 J=1,NP1                                                       IPT0340
 1320 NCON(I,J)=0                                                           IPT0350
      DO 1340 I=1,NNODE                                                     IPT0360
      F(I)=0.0D0                                                            IPT0370
```

```
      PHI(I)=0.0D0                                                      IPT0380
      PHIDOT(I)=0.0D0                                                   IPT0390
      PHIWRK(I)=0.0D0                                                   IPT0400
      DO 1340 J=1,3                                                     IPT0410
 1340 COORD(I,J)=0.0D0                                                  IPT0420
      DO 1360 I=1,NMAT                                                  IPT0430
      DO 1360 J=1,5                                                     IPT0440
 1360 PROP(I,J)=0.0D0                                                   IPT0450
      IF(NELBC.LE.0) GO TO 1390                                         IPT0460
      DO 1380 I=1,NELBC                                                 IPT0470
      PROPB(I)=0.0D0                                                    IPT0480
      KBCOND(I)=0                                                       IPT0490
      KSURF(I)=0                                                        IPT0500
 1380 KEL(I)=0                                                          IPT0510
C
C     READ THE MATERIAL PROPERTIES
C
 1390 DO 1400 I=1,NMAT                                                  IPT0520
      READ(5,1500) ID,(PROP(ID,J),J=1,3)                                IPT0530
      IF(IPROB.LE.1) GO TO 1400                                         IPT0540
      READ(5,1500) ID,PROP(ID,4),PROP(ID,5)                             IPT0550
 1400 CONTINUE                                                          IPT0560
 1500 FORMAT(I10,3F20.0)                                                IPT0570
      WRITE(6,1600)                                                     IPT0580
 1600 FORMAT(///,32X,' MATERIAL PROPERTIES',///,'  MTL# ',7X,' KX',18X,' IPT0590
     1KY',18X,'KZ',15X,'DENSITY',12X,'VISCOSITY',///)                   IPT0600
      DO 1700 I=1,NMAT                                                  IPT0610
 1700 WRITE(6,1800) I,(PROP(I,J),J=1,5)                                 IPT0620
 1800 FORMAT (I5,1P5G20.12,/)                                           IPT0630
C
C     READ THE JOINT COORDINATES
C     IF REQUIRED, READ BOUNDARY CONDITION NODAL VALUES AND/OR THE
C     INITIAL VALUES OF PHI AND PHIDOT
C
      DO 1850 I=1,NNODE                                                 IPT0640
 1850 READ(5,1875) IJT,(COORD(IJT,J),J=1,3)                             IPT0650
 1875 FORMAT(I10,3F20.0)                                                IPT0660
      IF(IPROB.LE.1) GO TO 2000                                         IPT0670
      DO 1900 I=1,NNODE                                                 IPT0680
 1900 READ(5,1950) IJT,PHI(IJT),PHIDOT(IJT)                             IPT0690
 1950 FORMAT(I10,2F20.0)                                                IPT0700
 2000 IF(NNBC.LE.0) GO TO 2075                                          IPT0710
      DO 2020 I=1,NNBC                                                  IPT0720
 2020 READ(5,2050) KNODE(I),PHIWRK(KNODE(I))                            IPT0730
 2050 FORMAT(I10,F20.0)                                                 IPT0740
      WRITE(6,2060)                                                     IPT0750
 2060 FORMAT(///,21X,' BOUNDARY CONDITION DATA',//,'  NODE',18X,'VALUE', IPT0760
     1///)                                                              IPT0770
```

```
      DO 2065 I=1,NNBC                                                  IPT0780
      ID=KNODE(I)                                                       IPT0790
 2065 WRITE(6,2070) ID,PHIWRK(ID)                                       IPT0800
 2070 FORMAT(I6,5X,1PG24.16)                                            IPT0810
 2075 WRITE(6,2100)                                                     IPT0820
 2100 FORMAT(///,' NODE #',10X,'X COORDINATE',12X,'Y COORDINATE',12X,'Z  IPT0830
     1COORDINATE',12X,'PHI INITIAL',9X,'PHIDOT INITIAL',///)            IPT0840
      DO 2200 I=1,NNODE                                                 IPT0850
 2200 WRITE(6,2300) I,(COORD(I,J),J=1,3),PHI(I),PHIDOT(I)               IPT0860
 2300 FORMAT(I5,5X,1P5G24.16)                                           IPT0870
C
C     IF COORDINATES NOT GIVEN IN RECTANGULAR SYSTEM, CONVERT
C
      IF(KC.GT.1) CALL MARK(COORD,KC,NNODE)                             IPT0880
C
C     READ ELEMENT CONNECTIVITY (2I5,23I3,/,9I3)
C
      DO 2400 I=1,NEL                                                   IPT0890
 2400 READ(5,2500) IEL,(NCON(IEL,J),J=1,NP1)                            IPT0900
 2500 FORMAT(2I5,23I3,/,9I3)                                            IPT0910
      WRITE (6,2600)                                                    IPT0920
 2600 FORMAT(//,21X,' CONNECTIVITY MATRIX',///,2X,' EL# MTL#',//)       IPT0930
      DO 2700 I=1,NEL                                                   IPT0940
 2700 WRITE(6,2800) I,(NCON(I,J),J=1,NP1)                               IPT0950
 2800 FORMAT(/,2I5,5X,20I4,/,15X,12I4)                                  IPT0960
C
C     COMPUTE THE HALF BAND WIDTH FROM CONNECTIVITY DATA
C
      NBAND=0                                                           IPT0970
      DO 3000 I=1,NEL                                                   IPT0980
      DO 2900 J=2,NPEL                                                  IPT0990
      JK=J+1                                                            IPT1000
      DO 2900 K=JK,NP1                                                  IPT1010
 2900 NBAND=MAX0(NBAND,IABS(NCON(I,J)-NCON(I,K)))                       IPT1020
 3000 CONTINUE                                                          IPT1030
      NBAND=NBAND+1                                                     IPT1040
      WRITE(6,3100) NBAND                                               IPT1050
 3100 FORMAT(///,11X,' SYSTEM HALF BAND WIDTH = ',I5,///)               IPT1060
C
C     READ IN BOUNDARY CONDITION DATA (3I5,F20.0)
C
      IF (NELBC.LE.0) GO TO 3700                                        IPT1070
      DO 3200 I=1,NELBC                                                 IPT1080
 3200 READ(5,3300) KEL(I),KBCOND(I),KSURF(I),PROPB(I)                   IPT1090
 3300 FORMAT(3I5,F20.0)                                                 IPT1100
      WRITE(6,3400)                                                     IPT1110
 3400 FORMAT(21X,' BOUNDARY CONDITION DATA',///,'  EL # BC # SURF       IPT1120
     1PROPERTY VALUE   ',///)                                           IPT1130
```

```
      DO 3500 I=1,NELBC                                                  IPT1140
 3500 WRITE(6,3600) KEL(I),KBCOND(I),KSURF(I),PROPB(I)                   IPT1150
 3600 FORMAT(' ',3I5,1PG24.16)                                           IPT1160
C
C     READ THE TRANSIENT PROBLEM DATA (I10,2F20.0)
C
 3700 IF(IPROB.LE.1) GO TO 4000                                          IPT1170
      READ(5,3725) NTSC,KINT,TIM                                         IPT1180
 3725 FORMAT(2I10,10X,F20.0)                                             IPT1190
      IF(NTSC.LE.1) NTSC=1                                               IPT1200
      DO 3750 I=1,NTSC                                                   IPT1210
 3750 READ(5,3800) IP(I),DTIM(I),ENDT(I)                                 IPT1220
 3800 FORMAT(I10,2F20.0)                                                 IPT1230
      IF(KINT.LT.1.OR.KINT.GT.3) KINT=3                                  IPT1240
      WRITE(6,3850) KINT,TIM                                             IPT1250
 3850 FORMAT(///,21X,' INTEGRATION TYPE NUMBER =',I5,///,21X,' START PRO IPT1260
     1BLEM AT TIME ',1PG20.8,///,21X,' TIME INTEGRATION PARAMETERS',//,' IPT1270
     2 SEGEMENT',6X,'DELTA TIME',15X,'END TIME',15X,'PRINT SKIP',//)     IPT1280
      DO 3900 I=1,NTSC                                                   IPT1290
 3900 WRITE(6,3950) I,DTIM(I),ENDT(I),IP(I)                              IPT1300
 3950 FORMAT(I10,1P2G25.16,I10)                                          IPT1310
 4000 RETURN                                                             IPT1320
      END                                                                IPT1330
```

```
      SUBROUTINE MERGE(IMTRX)                                                  MRG0010
      IMPLICIT REAL*8 (A-H,O-Z)                                                MRG0020
      COMMON /MTRX/ CORD(32,3),ENN(32,32),DCOL(32,3),COL(32),AJ(3,3),AJI       MRG0030
     1(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COARD(110       MRG0040
     2,3),F(110),PHI(110),PHIDOT(110),PHIWRK(110),PHIDDT(110),DTIM(5),EN       MRG0050
     3DT(5),BGH(110,64),BGC(110,64)                                            MRG0051
      COMMON /FLP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM                 MRG0060
      COMMON /INT/ IGO,NBAND,NNODE,NMAT,NEL,NPEL,NGP,NELBC,NNBC,IPROB,KC       MRG0070
     1,NTSC,KINT,IPJN                                                          MRG0071
      COMMON /IMTRX/ NCON(48,33),KEL(48),KBCOND(48),KSURF(48),KNODE(110)       MRG0080
     1,IP(5)                                                                   MRG0081
C
C     ZERO THE MASTER MATRIX
C
      DO 1000 I=1,NNODE                                                        MRG0090
      DO 1000 J=1,NBAND                                                        MRG0100
 1000 BGM(I,J)=0.0D0                                                           MRG0110
      DO 8000 I=1,NEL                                                          MRG0120
C
C     ZERO THE ELEMENT WORKING MATRICES
C
      DO 1100 II=1,NPEL                                                        MRG0130
      DO 1110 JJ=1,3                                                           MRG0140
 1110 CORD(II,JJ)=0.0D0                                                        MRG0150
      DO 1100 JI=1,NPEL                                                        MRG0160
 1100 ENW(II,JI)=0.0D0                                                         MRG0170
      DO 1200 J=1,NPEL                                                         MRG0180
      INDX=NCON(I,J+1)                                                         MRG0190
      DO 1200 K=1,3                                                            MRG0200
 1200 CORD(J,K)=COARD(INDX,K)                                                  MRG0210
      IMAT=NCON(I,1)                                                           MRG0220
      GO TO (2000,3000,4000),IMTRX                                             MRG0230
C
C     FORMING THE ELEMENTAL "H" MATRIX
C
 2000 AKX=PROP(IMAT,1)                                                         MRG0240
      AKY=PROP(IMAT,2)                                                         MRG0250
      AKZ=PROP(IMAT,3)                                                         MRG0260
      CALL CUBE(NPEL,NGP,1)                                                    MRG0270
      GO TO 6000                                                               MRG0280
C
C     FORMING THE ELEMENTAL "C" MATRIX
C
 3000 FACT=PROP(IMAT,5)                                                        MRG0290
      GO TO 4500                                                               MRG0300
C
```

```
C       FORMING THE ELEMENTAL "G" MATRIX                                        
C                                                                               
 4000 FACT=PROP(IMAT,4)                                                         MRG0310
 4500 CALL CUBE(NPEL,NGP,2)                                                     MRG0320
      DO 5000 II=1,NPEL                                                         MRG0330
      DO 5000 JJ=1,NPEL                                                         MRG0340
 5000 ENW(II,JJ)=FACT*ENW(II,JJ)                                                MRG0350
C                                                                               
C       ASSEMBLING THE MASTER MATRIX                                            
C                                                                               
 6000 DO 7000 II=1,NPEL                                                         MRG0360
      IDX=NCON(I,II+1)                                                          MRG0370
      DO 7000 JJ=1,NPEL                                                         MRG0380
      IDY=NCON(I,JJ+1)-IDX+1                                                    MRG0390
      IF (IDY.LT.1) GO TO 7000                                                  MRG0400
      BGM(IDX,IDY)=BGM(IDX,IDY)+ENW(II,JJ)                                      MRG0410
 7000 CONTINUE                                                                  MRG0420
 8000 CONTINUE                                                                  MRG0430
C                                                                               
C       SKIP "BCOND" IF THERE ARE NO ELEMENTS WITH BOUNDARY CONDITIONS          
C                                                                               
      IF(NELBC.GT.0.AND.IMTRX.EQ.1) CALL BCOND                                  MRG0440
C                                                                               
C       OUTPUT MATRIX                                                           
C                                                                               
      IF(IMTRX.EQ.1.AND.IPROB.NE.1) CALL STUFF(BGM,BGH,NNODE,NBAND)             MRG0450
      IF(IMTRX.EQ.2.AND.IPROB.EQ.3) CALL STUFF(BGM,BGC,NNODE,NBAND)             MRG0460
      RETURN                                                                    MRG0470
      END                                                                       MRG0480
```

```
      SUBROUTINE BCOND                                                  BCD0010
      IMPLICIT REAL*8(A-H,O-Z)                                          BCD0020
      COMMON /MTRX/ CORD(32,3),ENN(32,32),DCOL(32,3),COL(32),AJ(3,3),AJI BCD0030
     1(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COARD(110 BCD0040
     2,3),F(110),PHI(110),PHIDOT(110),PHIWRK(110),PHIDDT(110),DTIM(5),EN BCD0050
     3DT(5),BGH(110,64),BGC(110,64)                                     BCD0051
      COMMON /FLP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM          BCD0060
      COMMON /INT/ IGO,NBAND,NNODE,NMAT,NEL,NPEL,NGP,NELBC,NNBC,IPROB,KC BCD0070
     1,NTSC,KINT,IPUN                                                   BCD0071
      COMMON /IMTRX/ NCON(48,33),KEL(48),KBCOND(48),KSURF(48),KNODE(110) BCD0080
     1,IP(5)                                                            BCD0081
C
CC    FORM THE "F" VECTOR
C
      DO 800 I=1,NELBC                                                  BCD0090
C
CC    ZERO THE ELEMENT WORKING MATRICES
C
      DO 200 II=1,NPEL                                                  BCD0100
      FE(II)=0.0D0                                                      BCD0110
      DO 150 JJ=1,3                                                     BCD0120
  150 CORD(II,JJ)=0.0D0                                                 BCD0130
      DO 200 KK=1,NPEL                                                  BCD0140
  200 ENW(II,KK)=0.0D0                                                  BCD0150
C
CC    EXTRACT ELEMENT BOUNDARY CONDITION DATA
C
      IEL=KEL(I)                                                        BCD0160
      KBT=KBCOND(I)                                                     BCD0170
      KBS=KSURF(I)                                                      BCD0180
      CONST=PROPB(I)                                                    BCD0190
C
CC    EXTRACT ELEMENT COORDINATES
C
      DO 300 J=1,NPEL                                                   BCD0200
      INDX=NCON(IEL,J+1)                                                BCD0210
      DO 300 K=1,3                                                      BCD0220
  300 CORD(J,K)=COARD(INDX,K)                                           BCD0230
      CALL CUBE8(NPEL,NGP,KBT,KBS)                                      BCD0240
      GO TO (400,400,600),KBT                                           BCD0250
C
CC    ASSEMBLE TYPE 1 OR TYPE 2 BOUNDARY CONDITION
C
  400 DO 500 L=1,NPEL                                                   BCD0260
      ID=NCON(IEL,L+1)                                                  BCD0270
  500 F(ID)=F(ID)+CONST*FE(L)                                           BCD0280
```

```
      GO TO 800                                                          BCD0290
C
CC    ASSEMBLE TYPE 3 BOUNDARY CONDITION
C
  600 DO 650 II=1,NPEL                                                   BCD0300
      DO 650 JJ=1,NPEL                                                   BCD0310
  650 ENW(II,JJ)=CONST*ENW(II,JJ)                                        BCD0320
C
CC    ASSEMBLE BOUNDARY MATRIX INTO MASTER "H" MATRIX
C
      DO 700 KK=1,NPEL                                                   BCD0330
      IDX=NCON(IEL,KK+1)                                                 BCD0340
      DO 700 LL=1,NPEL                                                   BCD0350
      IDY=NCON(IEL,LL+1)-IDX+1                                           BCD0360
      IF (IDY.LT.1) GO TO 700                                            BCD0370
      BGM(IDX,IDY)=BGM(IDX,IDY)+ENW(KK,LL)                               BCD0380
  700 CONTINUE                                                           BCD0390
  800 CONTINUE                                                           BCD0400
      RETURN                                                             BCD0410
      END                                                                BCD0420
```

```
      SUBROUTINE STEADY                                                 SDY0010
      IMPLICIT REAL*8(A-H,O-Z)                                          SDY0020
      COMMON /MTRX/ CORD(32,3),ENN(32,32),DCOL(32,3),COL(32),AJ(3,3),AJI SDY0030
     1(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COORD(110 SDY0040
     2,3),F(110),PHI(110),PHIDOT(110),PHIWRK(110),PHIDDT(110),DTIM(5),EN SDY0050
     3DT(5),BGH(110,64),BGC(110,64)                                     SDY0051
      COMMON /FLP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM          SDY0060
      COMMON /INT/ IGO,NBAND,NNODE,NMAT,NEL,NPEL,NGP,NELBC,NNBC,IPROB,KC SDY0070
     1,NTSC,KINT,IPJN                                                   SDY0071
      COMMON /IMTRX/ NCON(48,33),KEL(48),KBCOND(48),KSURF(48),KNODE(110) SDY0080
     1,IP(5)                                                            SDY0081
C
CC    SOLVING THE STEADY STATE PROBLEM OF THE FORM (H)*<PHI>+<F>=0
CC    USING SUBROUTINES "LDLT" & "SLV"
CC
CCC
CC    APPLY THE SPECIFIED NODAL VALUES,PLACE "F" ON RIGHT HAND SIDE
CC    STORE "F" FOR REFERENCE
C
      DO 100 I=1,NNODE                                                  SDY0090
      PHI(I)=PHIWRK(I)                                                  SDY0091
      PHIWRK(I)=F(I)                                                    SDY0100
  100 F(I)=-F(I)                                                        SDY0110
C
CC    FIND THE APPROXIMATE VALUE OF ZERO
C
      CALL BNSN(BGM,NNODE,APZRO,ABIGN)
      WRITE(6,150) APZRO,ABIGN                                          SDY0130
  150 FORMAT('0',10X,'THE APPROXIMATE VALUE OF ZERO = ',1PG24.16,//,23X, SDY0140
     1' ABIGN = ',G24.16,//)                                            SDY0150
      IF(NNBC.LE.0) GO TO 450                                           SDY0160
      NEND=NBAND-1                                                      SDY0170
      DO 400 I=1,NNBC                                                   SDY0180
      ID=KNODE(I)                                                       SDY0190
      NCHK=NNODE-ID                                                     SDY0200
      DO 300 J=1,NEND                                                   SDY0210
      IROW=ID-J                                                         SDY0220
      ICOL=J+1                                                          SDY0230
      INDX=ID+J                                                         SDY0240
      IF(IROW.LE.0) GO TO 200                                           SDY0250
      F(IROW)=F(IROW)-BGM(IROW,ICOL)*PHI(ID)                            SDY0260
  200 IF(J.GT.NCHK) GO TO 300                                           SDY0270
      F(INDX)=F(INDX)-BGM(ID,ICOL)*PHI(ID)                              SDY0280
  300 CONTINUE                                                          SDY0290
      F(ID)=F(ID)-BGM(ID,1)*PHI(ID)                                     SDY0300
      BGM(ID,1)=ABIGN                                                   SDY0310
```

```
  400 CONTINUE                                                          SDY0320
  450 CALL LDLT(BGM,NNODE,NBAND,APZRO)                                  SDY0330
      CALL SLV(BGM,F,NNODE,NBAND)                                       SDY0340
C
C     FORM THE "PHI" AND "F" VECTORS
C
      IF(NNBC.LE.0) GO TO 600                                           SDY0350
      DO 500 I=1,NNBC                                                   SDY0360
      ID=KNODE(I)                                                       SDY0370
      PHIWRK(ID)=PHIWRK(ID)+ABIGN*F(ID)                                 SDY0380
  500 F(ID)=PHI(ID)                                                     SDY0390
  600 WRITE(6,800)                                                      SDY0400
  800 FORMAT('1',' NODE',7X,'X COORDINATE',8X,'Y COORDINATE',8X,'Z COORD SDY0410
     1INATE',4X,'PHI(STEADY STATE)',6X,'FORCING VECTOR',/,71X,' ADJ TO R SDY0420
     2EF',///)                                                          SDY0430
      DO 900 I=1,NNODE                                                  SDY0440
      PHI(I)=F(I)+PHIREF                                                SDY0450
  900 WRITE(6,1000) I,(COORD(I,J),J=1,3),PHI(I),PHIWRK(I)               SDY0460
 1000 FORMAT(I5,1X,1P3G20.5,2G20.10)                                    SDY0470
      TOT=0.0D0                                                         SDY0480
      DO 1100 I=1,NNODE                                                 SDY0490
 1100 TOT=TOT+PHIWRK(I)                                                 SDY0500
      WRITE(6,1200) TOT                                                 SDY0510
 1200 FORMAT(///,11X,' SUM OF FORCING VECTOR ELEMENTS IS',1PG20.10,/)   SDY0520
      IF(IPUN.GE.1) CALL PUNC(F,PHI,NNODE,IPROB)                        SDY0521
      RETURN                                                            SDY0530
      END                                                               SDY0540
```

```
      SUBROUTINE TIME11                                                 T110010
C
CC    THIS SUBROUTINE EMPLOYS THE EULER ONE METHOD
C
      IMPLICIT REAL*8(A-H,O-Z)                                          T110020
      COMMON /MTRX/ CORD(32,3),ENN(32,32),DCOL(32,3),COL(32),AJ(3,3),AJI T110030
     1(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COARD(110 T110040
     2,3),F(110),PHI(110),PHIDOT(110),PHIWRK(110),PHIDDT(110),DTIM(5),EN T110050
     3DT(5),BGH(110,64)                                                 T110060
      COMMON /FLP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM          T110070
      COMMON /INT/ IGO,NBAND,NNODE,NMAT,NEL,NPEL,NGP,NELBC,NNBC,IPROB,KC T110080
     1,NTSC,KINT,IPUN                                                   T110090
      COMMON /IMTRX/ NCON(48,33),KEL(48),KBCOND(48),KSURF(48),KNODE(110) T110100
     1,IP(5)                                                            T110110
      DIMENSION OUT(110,6),TIME(6)                                      T110120
      EQUIVALENCE (ENW(1,1),OUT(1,1))                                   T110130
C
CC    PREPARE THE "C" MATRIX FOR THE SOLUTION PROCESS
CC    INSERT BOUNDARY VALUE INTO PHI IF REQUIRED
C
      CALL BNSN(BGM,NNODE,APZRO,ABIGN)                                  T110140
      WRITE(6,100) APZRO,ABIGN                                          T110150
  100 FORMAT(///,11X,'THE APPROXIMATE VALUE OF ZERO =',1PG24.16,//,23X,' T110160
     1 ABIGN =',G24.16,//)                                              T110170
      IF(NNBC.LE.0) GO TO 400                                           T110180
      DO 300 I=1,NNBC                                                   T110190
      ID=KNODE(I)                                                       T110200
      PHI(ID)=PHIWRK(ID)                                                T110210
      BGM(ID,1)=ABIGN                                                   T110220
  300 CONTINUE                                                          T110230
  400 CALL LDLT(BGM,NNODE,NBAND,APZRO)                                  T110240
      WRITE(6,500)                                                      T110250
  500 FORMAT('1',40X,'DISTRIBUTION OF PHI(ADJ TO REF) VS TIME ',//,52X,' T110260
     1EULER I METHOD',//)                                               T110270
      DO 600 I=1,NNODE                                                  T110280
  600 OUT(I,1)=PHI(I)+PHIREF                                            T110290
C
CC    PREPARE DATA FOR TIME INTEGRATION PROCESS
C
      TIME(1)=0.0D0+TIM                                                 T110300
      IDP=1                                                             T110310
      DO 1400 MRH=1,NTSC                                                T110320
      JE=6                                                              T110330
      IXX=-1                                                            T110340
      DT=DTIM(MRH)                                                      T110350
      TI=(ENDT(MRH)-TIM)/DT                                             T110360
```

```
      IPRT=IP(MR8)                                                      T110370
      ITIM=TI+1.5                                                       T110380
      NTP=(ITIM-1)/IPRT                                                 T110390
      IF(MRH.EQ.1) NTP=NTP+1                                            T110400
C
C     SOLUTION ITERATION LOOP
C
      DO 1300 IT=1,ITIM                                                 T110410
      CALL MULT(BGH,PHI,PHIDOT,NNODE,NBAND)                             T110420
      DO 700 I=1,NNODE                                                  T110430
  700 PHIDOT(I)=-PHIDOT(I)-F(I)                                         T110440
      CALL SLV(BGM,PHIDOT,NNODE,NBAND)                                  T110450
      IF(NNBC.LE.0) GO TO 775                                           T110460
      DO 750 I=1,NNBC                                                   T110470
      ID=KNODE(I)                                                       T110480
      F(ID)=F(ID)+ABIGN*PHIDOT(ID)                                      T110490
  750 PHIDOT(ID)=0.0D0                                                  T110500
  775 DO 800 J=1,NNODE                                                  T110510
  800 PHI(J)=PHI(J)+DT*PHIDOT(J)                                        T110520
      IXX=IXX+1                                                         T110530
      IF(IXX.NE.IPRT) GO TO 1300                                        T110540
      IXX=0                                                             T110550
      IDP=IDP+1                                                         T110560
      TIME(IDP)=DT*(IT-1)+TIM                                           T110570
      DO 900 I=1,NNODE                                                  T110580
  900 OUT(I,IDP)=PHI(I)+PHIREF                                          T110590
C
C     WHEN OUT IS FULL, OUTPUT THE COLLECTED DATA
C
      IF(NTP.LT.JE) JE=NTP                                              T110600
      IF(IDP.LT.JE) GO TO 1300                                          T110610
      WRITE(6,1000) DT,(TIME(J),J=1,JE)                                 T110620
 1000 FORMAT(///,70X,'TIME OF PHI DISTRIBUTION',//,70X,'DELTA TIME =',F2 T110630
     10.5,//,41X,1P6G15.6)                                              T110640
      WRITE (6,1050)                                                    T110650
 1050 FORMAT (//,' NODE',2X,'X CORD',6X,'Y CORD',6X,'Z CORD',2X,6(9X,'PH T110660
     1I',3X),//)                                                        T110670
      DO 1100 I=1,NNODE                                                 T110680
 1100 WRITE(6,1200) I,(COARD(I,J),J=1,3),(OUT(I,K),K=1,JE)              T110690
 1200 FORMAT(I5,1P3G12.4,6G15.7)                                        T110700
      IDP=0                                                             T110710
      NTP=NTP-JE                                                        T110720
 1300 CONTINUE                                                          T110730
      IF(IPUN.GE.1) CALL PUNC(PHI,PHIDOT,NNODE,IPROB)                   T110740
 1400 TIM=DT*(ITIM-1)+TIM                                               T110750
      RETURN                                                            T110760
      END                                                               T110770
```

```
      SUBROUTINE TIME13                                                 T130010
C
CC    THIS SUBROUTINE EMPLOYS THE TIME SHAPE FUNCTION (LINEAR) METHOD
C     (KINT = 3) OR TRAPEZOIDAL (FINITE DIFFERENCE) METHOD (KINT = 2)
C
      IMPLICIT REAL*8(A-H,O-Z)                                          T130020
      COMMON /MTRX/ CORD(32,3),ENN(32,32),DCOL(32,3),COL(32),AJ(3,3),AJI T130030
     1(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COARD(110 T130040
     2,3),F(110),PHI(110),PHIDOT(110),PHIWRK(110),PHIDDT(110),DTIM(5),EN T130050
     3DT(5),BGH(110,64),BGC(110,64)                                     T130060
      COMMON /FLP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM          T130070
      COMMON /INT/ IGO,NBAND,NNODE,NMAT,NEL,NPEL,NGP,NELBC,NNBC,IPROB,KC T130080
     1,NTSC,KINT,IPUN                                                   T130090
      COMMON /IMTRX/ NCON(48,33),KEL(48),KBCOND(48),KSURF(48),KNODE(110) T130100
     1,IP(5)                                                            T130110
      DIMENSION OUT(110,6),TIME(6)                                      T130120
      EQUIVALENCE(ENW(1,1),OUT(1,1))                                    T130130
C
CC    PREPARE DATA FOR THE TIME INTEGRATION PROCESS
CC    SAVE THE "F" VECTOR IN "PHIDDT" FOR RECOVERY AND REFERENCE
C
      DO 25 I=1,NNODE                                                   T130140
   25 PHIDDT(I)=F(I)                                                    T130150
      TIME(1)=0.0D0+TIM                                                 T130160
      IDP=1                                                             T130170
C
CC    LOOP TO VARY TIME STEP SIZES
C
      IF(NTSC.EQ.1) GO TO 50                                            T130180
      REWIND 10                                                         T130190
      REWIND 11                                                         T130200
      WRITE(10) BGM                                                     T130210
      WRITE(11) BGH                                                     T130220
   50 DO 1900 MRH=1,NTSC                                                T130230
      IF(MRH.EQ.1) GO TO 75                                             T130240
      REWIND 10                                                         T130250
      REWIND 11                                                         T130260
      READ(10) BGM                                                      T130270
      READ(11) BGH                                                      T130280
   75 JE=6                                                              T130290
      IXX=-1                                                            T130300
      DT=DTIM(MRH)                                                      T130310
      TI=(ENDT(MRH)-TIM)/DT                                             T130320
      IPRT=IP(MRH)                                                      T130330
      ITIM=TI+1.5                                                       T130340
      NTP=(ITIM-1)/IPRT                                                 T130350
```

```
      IF(MRH.EQ.1) NTP=NTP+1                                              T130360
C
CCC   COMBINE THE "H" AND "C" MATRICES FOR THE SOLUTION PROCESS
CCC   LEFT SIDE MATRIX IS "BGM" ; RIGHT SIDE MATRIX IS "BGH"
C
      IF(KINT.EQ.3) GO TO 125                                             T130370
      DO 100 I=1,NNODE                                                    T130380
      DO 100 J=1,NBAND                                                    T130390
      CIJ=BGM(I,J)/DT                                                     T130400
      AHIJ=BGH(I,J)/2.0D0                                                 T130410
      BGM(I,J)=AHIJ+CIJ                                                   T130420
  100 BGH(I,J)=AHIJ-CIJ                                                   T130430
      GO TO 175                                                           T130440
  125 DO 150 I=1,NNODE                                                    T130450
      DO 150 J=1,NBAND                                                    T130460
      CIJ=BGM(I,J)/DT                                                     T130470
      AHIJ=BGH(I,J)/3.0D0                                                 T130480
      BGM(I,J)=2.0D0*AHIJ+CIJ                                             T130490
  150 BGH(I,J)=AHIJ-CIJ                                                   T130500
C
CCC   PREPARE THE "LHS" MATRIX FOR THE SOULTION PROCESS
CCC   TO ACCOMODATE A TIME DEPENDENT "F" VECTOR, THE CORRECT ELEMENTS
CCC   FROM "BGM" MUST BE SAVED AND THE CORRECTION TO "F" APPLIED
CCC   IN THE TIME ITERATION LOOP
C
  175 CALL BNSN(BGM,NNODE,APZRO,ABIGN)                                    T130510
      WRITE(6,200) APZRO,ABIGN                                            T130520
  200 FORMAT(///,11X,'THE APPROXIMATE VALUE OF ZERO =',1PG24.16,//,23X,'  T130530
     1 ABIGN =',G24.16,//)                                                T130540
      IF(NNBC.LE.0) GO TO 600                                             T130550
      NEND=NBAND-1                                                        T130560
      DO 500 I=1,NNBC                                                     T130570
      ID=KNODE(I)                                                         T130580
      PHI(ID)=PHIWRK(ID)                                                  T130590
      NCHK=NNODE-ID                                                       T130600
      DO 400 J=1,NEND                                                     T130610
      IROW=ID-J                                                           T130620
      ICOL=J+1                                                            T130630
      INDX=ID+J                                                           T130640
      IF(IROW.LE.0) GO TO 300                                             T130650
      F(IROW)=F(IROW)+BGM(IROW,ICOL)*PHI(ID)                              T130660
  300 IF(J.GT.NCHK) GO TO 400                                             T130670
      F(INDX)=F(INDX)+BGM(ID,ICOL)*PHI(ID)                                T130680
  400 CONTINUE                                                            T130690
      F(ID)=F(ID)+BGM(ID,1)*PHI(ID)                                       T130700
  500 BGM(ID,1)=ABIGN                                                     T130710
  600 CALL LDLT(BGM,NNODE,NBAND,APZRO)                                    T130720
      IF(MRH.GT.1) GO TO 900                                              T130730
```

```
      WRITE(6,700)                                                      T130740
  700 FORMAT('1',40X,'DISTRIBUTION OF PHI(ADJ TO REF) VS TIME',//)      T130750
      IF(KINT.EQ.2) GO TO 740                                           T130760
      WRITE(6,720)                                                      T130770
  720 FORMAT(47X,'TIME SHAPE FUNCTION METHOD',//)                       T130780
      GO TO 780                                                         T130790
  740 WRITE(6,760)                                                      T130800
  760 FORMAT(51X,'TRAPEZOIDAL METHOD',//)                               T130810
  780 DO 800 I=1,NNODE                                                  T130820
  800 OUT(I,1)=PHI(I)+PHIREF                                            T130830
C
CC    SOLUTION ITERATION LOOP
C
  900 DO 1800 IT=1,ITIM                                                 T130840
      CALL MULT(BGH,PHI,PHIDOT,NNODE,NBAND)                             T130850
      DO 1000 I=1,NNODE                                                 T130860
 1000 PHI(I)=-PHIDOT(I)-F(I)                                            T130870
      CALL SLV(BGM,PHI,NNODE,NBAND)                                     T130880
      IF(NNBC.LE.0) GO TO 1200                                          T130890
      DO 1100 I=1,NNBC                                                  T130900
      ID=KNODE(I)                                                       T130910
      SAV=ABIGN*PHI(ID)                                                 T130920
      F(ID)=F(ID)+SAV                                                   T130930
      PHIDOT(ID)=PHIDOT(ID)+SAV                                         T130940
 1100 PHI(ID)=PHIWRK(ID)                                                T130950
 1200 IXX=IXX+1                                                         T130960
      IF(IXX.NE.IPRT) GO TO 1800                                        T130970
      IXX=0                                                             T130980
      IDP=IDP+1                                                         T130990
      TIME(IDP)=DT*(IT-1)+TIM                                           T131000
      DO 1300 I=1,NNODE                                                 T131010
 1300 OUT(I,IDP)=PHI(I)+PHIREF                                          T131020
C
CC    WHEN OUT IS FULL, OUTPUT THE COLLECTED DATA
C
      IF(NTP.LT.JE) JE=NTP                                              T131030
      IF(IDP.LT.JE) GO TO 1800                                          T131040
      WRITE(6,1400) DT,(TIME(J),J=1,JE)                                 T131050
 1400 FORMAT(///,70X,'TIME OF PHI DISTRIBUTION',//,70X,'DELTA TIME =',F2 T131060
     10.5,//,41X,1P6G15.6)                                              T131070
      WRITE (6,1500)                                                    T131080
 1500 FORMAT (//,' NODE',2X,'X CORD',6X,'Y CORD',6X,'Z CORD',2X,6(9X,'PH T131090
     1I',3X),//)                                                        T131100
      DO 1600 I=1,NNODE                                                 T131110
 1600 WRITE(6,1700) I,(COARD(I,J),J=1,3),(OUT(I,K),K=1,JE)              T131120
 1700 FORMAT(I5,1P3G12.4,6G15.7)                                        T131130
      NTP=NTP-JE                                                        T131140
      IDP=0                                                             T131150
```

```
 1800 CONTINUE                                                          T131160
C
C       RECOVER THE TRUE "F" VECTOR
C
                                                                        T131170
        DO 1850 I=1,NNODE                                               T131180
 1850   F(I)=PHIDDT(I)                                                  T131190
        IF(IPUN.GE.1) CALL PUNC(PHI,PHIDOT,NNODE,IPROB)                 T131200
 1900   TIM=DT*(ITIM-1)+TIM                                             T131210
        RETURN                                                          T131220
        END
```

```
      SUBROUTINE TIME21                                                 T210010
C
CC    THIS SUBROUTINE EMPLOYS THE EULER I METHOD
C
      IMPLICIT REAL*8(A-H,O-Z)                                          T210020
      COMMON /MTRX/ CORD(32,3),ENN(32,32),DCOL(32,3),COL(32),AJ(3,3),AJI T210030
     1(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COARD(110 T210040
     2,3),F(110),PHI(110),PHIDOT(110),PHIWRK(110),PHIDDT(110),DTIM(5),EN T210050
     3DT(5),BGH(110,64),BGC(110,64)                                     T210060
      COMMON /FLP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM          T210070
      COMMON /INT/ IGO,NBAND,NNODE,NMAT,NEL,NPEL,NGP,NELBC,NNBC,IPROB,KC T210080
     1,NTSC,KINT,IPJN                                                   T210090
      COMMON /IMTRX/ NCON(48,33),KEL(48),KBCOND(48),KSURF(48),KNODE(110) T210100
     1,IP(5)                                                            T210110
      DIMENSION OUT(110,6),TIME(3)                                      T210120
      EQUIVALENCE(ENW(1,1),OUT(1,1))                                    T210130
C
CC    PREPARE THE "G" MATRIX FOR THE SOLUTION PROCESS
CC    INSERT BOUNDARY VALUES INTO PHI IF REQUIRED
C
      CALL BNSN(BGM,NNODE,APZRO,ABIGN)                                  T210140
      WRITE(6,100) APZRO,ABIGN                                          T210150
  100 FORMAT(///,11X,'THE APPROXIMATE VALUE OF ZERO =',1PG24.16,//,23X,' T210160
     1 ABIGN =',G24.16,//)                                              T210170
      IF(NNBC.LE.0) GO TO 400                                           T210180
      DO 300 I=1,NNBC                                                   T210190
      ID=KNODE(I)                                                       T210200
      PHI(ID)=PHIWRK(ID)                                                T210210
      BGM(ID,1)=ABIGN                                                   T210220
  300 CONTINUE                                                          T210230
  400 CALL LDLT(BGM,NNODE,NBAND,APZRO)                                  T210240
      WRITE(6,800)                                                      T210250
  800 FORMAT('1',40X,'DISTRIBUTION OF PHI(ADJ TO REF) AND PHIDOT VS TIME T210260
     1',//,56X,'EULER ONE METHOD',//)                                   T210270
      DO 900 I=1,NNODE                                                  T210280
      OUT(I,1)=PHI(I)+PHIREF                                            T210290
  900 OUT(I,2)=PHIDOT(I)                                                T210300
      TIME(1)=0.0D0+TIM                                                 T210310
      IDP=1                                                             T210320
      DO 1900 MRH=1,NTSC                                                T210330
      JE=6                                                              T210340
      IXX=-1                                                            T210350
      DT=DTIM(MRH)                                                      T210360
      TI=(ENDT(MRH)-TIM)/DT                                             T210370
      IPRT=IP(MRH)                                                      T210380
      ITIM=TI+1.5                                                       T210390
```

```
      NTP=(ITIM-1)/IPRT*2                                                T210400
      IF(MRH.EQ.1) NTP=NTP+2                                             T210410
C
C     SOLUTION ITERATION LOOP
C
      DO 1800 IT=1,ITIM                                                  T210420
      CALL MULT(BGH,PHI,PHIDDT,NNODE,NBAND)                              T210430
      DO 1000 I=1,NNODE                                                  T210440
 1000 PHIDDT(I)=-PHIDDT(I)-F(I)                                          T210450
      IF(IPROB.EQ.4) GO TO 1200                                          T210460
      CALL MULT(BGC,PHIDOT,PHIWRK,NNODE,NBAND)                           T210470
      DO 1100 I=1,NNODE                                                  T210480
 1100 PHIDDT(I)=PHIDDT(I)-PHIWRK(I)                                      T210490
 1200 CALL SLV(BGM,PHIDDT,NNODE,NBAND)                                   T210500
      IF(NNBC.LE.0) GO TO 1250                                           T210510
      DO 1225 I=1,NNBC                                                   T210520
      ID=KNODE(I)                                                        T210530
      F(ID)=F(ID)+ABIGN*PHIDDT(ID)                                       T210540
 1225 PHIDDT(ID)=0.000                                                   T210550
 1250 DO 1300 J=1,NNODE                                                  T210560
      PHI(J)=PHI(J)+DT*PHIDOT(J)                                         T210570
 1300 PHIDOT(J)=PHIDOT(J)+DT*PHIDDT(J)                                   T210580
      IXX=IXX+1                                                          T210590
      IF(IXX.NE.IPRT) GO TO 1800                                         T210600
      IXX=0                                                              T210610
      IDP=IDP+1                                                          T210620
      TIME(IDP)=DT*(IT-1)+TIM                                            T210630
      IPD=2*IDP                                                          T210640
      LL=IPD-1                                                           T210650
      DO 1400 I=1,NNODE                                                  T210660
      OUT(I,LL)=PHI(I)+PHIREF                                            T210670
 1400 OUT(I,IPD)=PHIDOT(I)                                               T210680
C
C     WHEN "OUT" IS FULL, PRINT THE COLLECTED DATA
C
      IF(NTP.LT.JE) JE=NTP                                               T210690
      L=JE/2                                                             T210700
      IF(IDP.LT.L) GO TO 1800                                            T210710
      WRITE(6,1500) DT,(TIME(J),J=1,L)                                   T210720
 1500 FORMAT(///,70X,'TIME OF PHI AND PHIDOT DISTRIBUTION',//,70X,'DELTA T210730
     1  TIME =',F20.5,//,41X,3(8X,1PG14.6,8X))                          T210740
      WRITE(6,1550)                                                      T210750
 1550 FORMAT(//,' NODE',2X,'X CORD',6X,'Y CORD',6X,'Z CORD',2X,3(8X,'PHI T210760
     1',10X,'PHIDOT',3X),//)                                            T210770
      DO 1600 I=1,NNODE                                                  T210780
 1600 WRITE(6,1700) I,(COORD(I,J),J=1,3),(OUT(I,K),K=1,JE)               T210790
 1700 FORMAT(I5,1P3G12.4,6G15.7)                                         T210800
      IDP=0                                                              T210810
```

```
      NTP=NTP-JE                                                        T210820
 1800 CONTINUE                                                          T210830
      IF(IPUN.GE.1) CALL PUNC(PHI,PHIDOT,NNODE,IPROB)                   T210840
 1900 TIM=DT*(ITIM-1)+TIM                                               T210850
      RETURN                                                            T210860
      END                                                               T210870
```

```
      SUBROUTINE TIME23                                                  T230010
C
C     THIS SUBROUTINE EMPLOYS A FOUR POINT BACKWARD FINITE DIFFERENCE
C     FORMULAE.  STARTING VALUES ARE OBTAINED BY A TAYLOR SERIES
C     EXPANSION
C
      IMPLICIT REAL*8(A-H,O-Z)                                           T230019
      COMMON /MTRX/ CORD(32,3),ENN(32,32),DCOL(32,3),COL(32),AJ(3,3),AJI T230020
     1(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COARD(110 T230030
     2,3),F(110),PHI(110),PHIDOT(110),PHIWRK(110),PHIDDT(110),DTIM(5),EN T230040
     3DT(5),BGH(110,64),BGC(110,64)                                      T230050
      COMMON /FLP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM           T230060
      COMMON /INT/ IGO,NBAND,NNODE,NMAT,NEL,NPEL,NGP,NELBC,NNBC,IPROB,KC T230070
     1,NTSC,KINT,IPUN                                                    T230080
      COMMON /IMTRX/ NCON(48,33),KEL(48),KBCOND(48),KSURF(48),KNODE(110) T230090
     1,IP(5)                                                             T230100
      DIMENSION OUT(110,6),TIME(6),SAVE(110,3),FSAV(110),START(110,3),PH T230110
     1ISAV(110),PHID(110)                                                T230120
      EQUIVALENCE (ENW(1,1),OUT(1,1)),(ENN(1,1),SAVE(1,1)),(ENN(331),FSA T230130
     1V(1)),(ENN(441),START(1,1)),(ENN(771),PHISAV(1)) ,(ENW(881),PHID(1 T230140
     2))                                                                 T230150
C
C     OBTAIN THREE STARTING VALUES OF PHI FROM A FIVE TERM TAYLOR
C     SERIES EXPANSION
C
      REWIND 10                                                          T230160
      WRITE (10) BGM                                                     T230170
      REWIND 10                                                          T230180
      CALL BNSN(BGM,NNODE,APZRO,ABIGN)                                   T230190
      DT=DTIM(1)                                                         T230200
      IF (NNBC.LE.0) GO TO 200                                           T230210
      DO 100 I=1,NNBC                                                    T230220
      ID=KNODE(I)                                                        T230230
      PHI(ID)=PHIWRK(ID)                                                 T230240
  100 BGM(ID,1)=ABIGN                                                    T230250
  200 CALL LDLT(BGM,NNODE,NBAND,APZRO)                                   T230260
      DO 300 I=1,NNODE                                                   T230270
      START(I,1)=PHI(I)                                                  T230280
  300 FSAV(I)=PHIDOT(I)                                                  T230290
      DO 1700 ITAIL=1,2                                                  T230300
      ITP1=ITAIL+1                                                       T230310
      DO 400 I=1,NNODE                                                   T230320
      START(I,ITP1)=START(I,ITAIL)+DT*FSAV(I)                            T230330
      PHI(I)=START(I,ITAIL)                                              T230340
  400 PHIDOT(I)=FSAV(I)                                                  T230350
      DO 1700 J=1,5                                                      T230360
```

```
      TOP=DT**J                                                          T230370
      BOT=1.0D0                                                          T230380
      DO 500 KX=1,J                                                      T230390
  500 BOT=BOT*DFLOAT(KX)                                                 T230400
      CALL MULT(BGH,PHI,PHISAV,NNODE,NBAND)                              T230410
      IF(IPROB.EQ.4) GO TO 550                                           T230420
      CALL MULT(BGC,PHIDOT,PHIDDT,NNODE,NBAND)                           T230430
  550 IF(J.GT.1) GO TO 900                                               T230440
      IF(IPROB.EQ.4) GO TO 700                                           T230450
      DO 600 I=1,NNODE                                                   T230460
  600 PHI(I)=-F(I)-PHISAV(I)-PHIDDT(I)                                   T230470
      GO TO 900                                                          T230480
  700 DO 800 I=1,NNODE                                                   T230490
  800 PHI(I)=-F(I)-PHISAV(I)                                             T230500
  900 IF(IPROB.EQ.4) GO TO 1100                                          T230510
      DO 1000 I=1,NNODE                                                  T230520
 1000 PHI(I)=-PHISAV(I)-PHIDDT(I)                                        T230530
      GO TO 1300                                                         T230540
 1100 DO 1200 I=1,NNODE                                                  T230550
 1200 PHI(I)=-PHISAV(I)                                                  T230560
 1300 CALL SLV(BGM,PHI,NNODE,NBAND)                                      T230570
      IF(NNBC.LE.0) GO TO 1500                                           T230580
      DO 1400 I=1,NNBC                                                   T230590
      ID=KNODE(I)                                                        T230600
      IF(J.GT.1) GO TO 1400                                              T230610
      F(ID)=F(ID)+ABIGN*PHI(ID)                                          T230620
 1400 PHI(ID)=0.0D0                                                      T230630
 1500 DO 1600 I=1,NNODE                                                  T230640
      FSAV(I)=FSAV(I)+TOP*PHI(I)/BOT                                     T230650
      START(I,ITP1)=START(I,ITP1)+TOP*DT*PHI(I)/DFLOAT(J+1)              T230660
      YY=PHI(I)                                                          T230670
      PHI(I)=PHIDOT(I)                                                   T230680
 1600 PHIDOT(I)=YY                                                       T230690
 1700 CONTINUE                                                           T230700
      READ(10) BGM                                                       T230710
      DO 1800 I=1,NNODE                                                  T230720
      PHIDOT(I)=0.0D0                                                    T230730
 1800 PHI(I)=START(I,3)                                                  T230740
C
C     STARTING CALCULATIONS COMPLETE: STARTING VECTORS ARE IN
C     "START" MATRIX
C
C
C     PREPARE DATA FOR THE TIME INTEGRATION PROCESS, SAVE THE "F"
C     VECTOR FOR RECOVERY AND REFERENCE
C
      DO 2000 I=1,NNODE                                                  T230750
 2000 FSAV(I)=F(I)                                                       T230760
```

```
      TIME(1)=0.0D0+TIM                                                 T230770
      IDP=1                                                             T230780
      DTSAV=1.0D0                                                       T230790
      REWIND 10                                                         T230800
      WRITE(10) BGH                                                     T230810
C
C     LOOP TO VARY TIME STEP SIZE
C
      DO 5300 MRH=1,NTSC                                                T230850
      DT=DTIM(MRH)                                                      T230860
      TI=(ENDT(MRH)-TIM)/DT                                             T230870
      IPRT=IP(MRH)                                                      T230880
      NTP=TI/IPRT                                                       T230890
      IF(MRH.EQ.1) GO TO 2100                                           T230900
      REWIND 10                                                         T230910
      READ(10) BGH                                                      T230920
      IXX=-1                                                            T230930
      IS=1                                                              T230940
      GO TO 2200                                                        T230950
 2100 IS=3                                                              T230960
      IXX=1                                                             T230970
      NTP=NTP+1                                                         T230980
 2200 JE=6                                                              T230990
      ITIM=TI+1.5                                                       T231000
      TOP=DTSAV*DTSAV                                                   T231010
      BOT=DT*DT                                                         T231020
C
C     COMBINE THE "H", "C", & "G" MATRICIES FOR THE SOLUTION PROCESS
C     LEFT SIDE MATRIX IS "BGH"; RIGHT SIDE MATRICES ARE "BGC" AND "BGM"
C     REMOVE THE DELTA TIME CONTRIBUTION FROM "BGC" AND "BGM" IF
C     NECESSARY
C
      IF(IPROB.EQ.3) GO TO 2400                                         T231030
      DO 2300 I=1,NNODE                                                 T231040
      DO 2300 J=1,NBAND                                                 T231050
      BGM(I,J)=TOP*BGM(I,J)/BOT                                         T231060
 2300 BGH(I,J)=BGH(I,J)+2.0D0*BGM(I,J)                                  T231070
      GO TO 2600                                                        T231080
 2400 DO 2500 I=1,NNODE                                                 T231090
      DO 2500 J=1,NBAND                                                 T231100
      BGM(I,J)=TOP*BGM(I,J)/BOT                                         T231110
      BGC(I,J)=DTSAV*BGC(I,J)/DT                                        T231120
 2500 BGH(I,J)=BGH(I,J)+2.0D0*BGM(I,J)+3.0D0*BGC(I,J)/2.0D0             T231130
C
C     PREPARE THE "LHS" MATRIX AND "F" VECTOR FOR THE SOLUTION PROCESS
C
 2600 CALL BNSN(BGH,NNODE,APZRO,ABIGN)                                  T231140
      WRITE(6,2900) APZRO,ABIGN                                         T231150
```

```
 2900 FORMAT(///,11X,'THE APPROXIMATE VALUE OF ZERO =',1PG24.16,//,23X,' T231160
     1ABIGN =',G24.16,//)                                                T231170
      IF (NNBC.LE.0) GO TO 3300                                          T231180
      NEND=NBAND-1                                                       T231190
      DO 3200 I=1,NNBC                                                   T231200
      ID=KNODE(I)                                                        T231210
      PHI(ID)=PHIWRK(ID)                                                 T231230
      NCHK=NNODE-ID                                                      T231240
      DO 3100 J=1,NEND                                                   T231250
      IROW=ID-J                                                          T231260
      INDX=ID+J                                                          T231270
      ICOL=J+1                                                           T231280
      IF(IROW.LE.0) GO TO 3000                                           T231290
      F(IROW)=F(IROW)+BGH(IROW,ICOL)*PHI(ID)                             T231300
 3000 IF(J.GT.NCHK) GO TO 3100                                           T231310
      F(INDX)=F(INDX)+BGH(ID,ICOL)*PHI(ID)                               T231320
 3100 CONTINUE                                                           T231330
      F(ID)=F(ID)+BGH(ID,1)*PHI(ID)                                      T231340
 3200 BGH(ID,1)=ABIGN                                                    T231350
 3300 CALL LDLT(BGH,NNODE,NBAND,APZRO)                                   T231360
      IF(MRH.GT.1) GO TO 3700                                            T231370
      WRITE(6,3400)                                                      T231380
 3400 FORMAT('1',40X,'DISTRIBUTION OF PHI(ADJ TO REF) VS TIME',//,41X,'F T231390
     1OUR POINT BACKWARD DIFFERENCE METHOD',//)                          T231400
      DO 3500 I=1,NNODE                                                  T231410
 3500 OUT(I,1)=START(I,1)+PHIREF                                         T231420
      IF(IPRT.GE.3) GO TO 3700                                           T231430
      DO 3600 LLL=2,3,IPRT                                               T231440
      MMM=LLL-1                                                          T231450
      IF(IPRT.EQ.2) MMM=IPRT                                             T231460
      IXX=IXX+1                                                          T231470
      IDP=IDP+1                                                          T231480
      TIME(IDP)=DT*MMM+TIM                                               T231490
      DO 3600 I=1,NNODE                                                  T231500
 3600 OUT(I,LLL)=START(I,LLL)+PHIREF                                     T231510
C
C     OBTAIN THE INDICES OF VALUES TO SAVE FOR VARYING TIME STEPS
C
 3700 IF(MRH.EQ.NTSC) GO TO 3800                                         T231520
      RAT=DTIM(MRH+1)/DT                                                 T231530
      KSAV=RAT                                                           T231540
      IM2S=ITIM-2*KSAV                                                   T231550
      IM3S=ITIM-3*KSAV                                                   T231560
C
C     STARTING VECTORS
C
 3800 DO 3900 I=1,NNODE                                                  T231570
      DO 3900 J=1,3                                                      T231580
```

```
 3900 SAVE(I,J)=START(I,J)                                                T231590
      IM3=1                                                               T231600
      IM2=2                                                               T231610
      IM1=3                                                               T231620
C
C     TIME ITERATION LOOP
C
      DO 5100 IT=IS,ITIM                                                  T231630
      DO 4000 I=1,NNODE                                                   T231640
 4000 PHIDDT(I)=-SAVE(I,IM3)+4.0D0*SAVE(I,IM2)-5.0D0*SAVE(I,IM1)          T231650
      IF(IPROB.EQ.4) GO TO 4200                                           T231660
      DO 4100 I=1,NNODE                                                   T231670
 4100 PHISAV(I)=SAVE(I,IM2)/2.0D0-2.0D0*SAVE(I,IM1)                       T231680
      CALL MULT(BGC,PHISAV,PHIDOT,NNODE,NBAND)                            T231690
 4200 CALL MULT(BGM,PHIDDT,PHI,NNODE,NBAND)                               T231700
      DO 4300 I=1,NNODE                                                   T231710
 4300 PHI(I)=-PHI(I)-F(I)-PHIDOT(I)                                       T231720
      CALL SLV(BGH,PHI,NNODE,NBAND)                                       T231730
      IF(NNBC.LE.0) GO TO 4500                                            T231740
      DO 4400 I=1,NNBC                                                    T231750
      ID=KNODE(I)                                                         T231760
      COR=ABIGN*PHI(ID)                                                   T231770
      F(ID)=F(ID)+COR                                                     T231780
      FSAV(ID)=FSAV(ID)+COR                                               T231790
 4400 PHI(ID)=PHIWRK(ID)                                                  T231800
 4500 CALL STUFFV(SAVE,PHI,NNODE,IM3)                                     T231810
      IXX=IXX+1                                                           T231820
      IM3=IM3+1                                                           T231830
      IM2=IM2+1                                                           T231840
      IM1=IM1+1                                                           T231850
      IF(IM3.GT.3) IM3=1                                                  T231860
      IF(IM2.GT.3) IM2=1                                                  T231870
      IF(IM1.GT.3) IM1=1                                                  T231890
      IF(IT.EQ.IM3S) CALL STUFFV(START,PHI,NNODE,1)                       T231900
      IF(IT.EQ.IM2S) CALL STUFFV(START,PHI,NNODE,2)                       T231910
      IF(IXX.NE.IPRT) GO TO 5100                                          T231920
      IXX=0                                                               T231930
      IDP=IDP+1                                                           T231940
      TIME(IDP)=DT*(IT-1)+TIM                                             T231950
      DO 4700 I=1,NNODE                                                   T231960
 4700 OUT(I,IDP)=PHI(I)+PHIREF                                            T231970
C
C     IF "OUT" IS FULL OUTPUT THE COLLECTED DATA
C
      IF(NTP.LT.JE) JE=NTP                                                T231980
      IF(IDP.LT.JE) GO TO 5100                                            T231990
      WRITE(6,4800) DT,(TIME(J),J=1,JE)                                   T232000
 4800 FORMAT(///,70X,'TIME OF PHI DISTRIBUTION',//,70X,'DELTA TIME =',F2  T232010
```

```
     10.5,//,42X,1P6G15.6)                                                    T232020
      WRITE (6,4850)                                                         T232030
 4850 FORMAT (//,' NODE',2X,'X CORD',6X,'Y CORD',6X,'Z CORD',4X,6(9X,'PH     T232040
     1I',3X),//)                                                             T232050
      DO 4900 I=1,NNODE                                                      T232060
 4900 WRITE(6,5000) I,(COARD(I,J),J=1,3),(OUT(I,K),K=1,JE)                   T232070
 5000 FORMAT(I5,1P3G12.4,6G15.7)                                             T232080
      NTP=NTP-JE                                                             T232090
      IDP=0                                                                  T232100
 5100 CONTINUE                                                               T232110
      CALL STUFFV(START,PHI,NNODE,3)                                         T232220
C
C     RECOVER THE TRUE "F" VECTOR AND "PHIDOT"
C
      DO 5200 I=1,NNODE                                                      T232230
      PHID(I)=(START(I,2)-4.0D0*START(I,3)+3.0D0*PHI(I))/DT/2.0D0            T232240
 5200 F(I)=FSAV(I)                                                           T232250
      DTSAV=DT                                                               T232260
      IF(IPUN.GE.1) CALL PUNC(PHI,PHID,NNODE,IPROB)                          T232270
 5300 TIM=DT*(ITIM-1)+TIM                                                    T232280
      RETURN                                                                 T232290
      END                                                                    T232300
```

```
      SUBROUTINE CUBE(NPEL,NGP,ITYPE)                                   CUB0010
      IMPLICIT REAL*8 (A-H,O-Z)                                         CUB0020
      COMMON /MTRX/ CORD(32,3),ENN(32,32),DCOL(32,3),COL(32),AJ(3,3),AJI CUB0030
     1(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COORD(110 CUB0040
     2,3),F(110),PHI(110),PHIDOT(110),PHIWRK(110),PHIDDT(110),DTIM(5),EN CUB0050
     3DT(5),BGH(110,64),BGC(110,64)                                     CUB0051
      COMMON /FLP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM          CUB0060
      DIMENSION GP(6,5),WF(6,5)                                         CUB0070
      DATA GP/.57735026918963D0,-.57735026918963D0,0.0D0,0.0D0,0.0D0,0.D CUB0080
     10,.77459666924148D0,0.0D0,-.77459666924148D0,0.0D0,0.0D0,0.0D0,.86 CUB0090
     211363115940SD0,0.33998104358486D0,-.33998104358486D0,-.86113631159 CUB0100
     3405D0,0.0D0,0.0D0,.90617984593866D0,.53846931010568D0,0.0D0,-.5384 CUB0110
     46931010568D0,-.90617984593866D0, 0.0D0,.93246951420315D0,.66120938 CUB0120
     5646626D0,.23861918608317D0,-.23861918608317D0,-.66120938646626D0,- CUB0130
     6.93246951420315D0/                                                CUB0140
      DATA WF/1.0D0,1.0D0,0.0D0,0.0D0,0.0D0,0.0D0,.55555555555556D0,.888 CUB0150
     188888888889D0,.55555555555556D0,0.0D0,0.0D0,0.0D0,.34785484513745D CUB0160
     20,.65214515486255D0,.65214515486255D0,.34785484513745D0,0.0D0,0.0D CUB0170
     3D,.23692688505619D0,.47862867049937D0,.56888888888889D0,.478628670 CUB0180
     449937D0,.23692688505619D0,0.0D0,.17132449237917D0,.36076157048140 CUB0190
     50,.46791393457269D0,.46791393457269D0,.36076157304814D0,.171324492 CUB0200
     637917D0/                                                          CUB0210
C
CC    CHECK FOR AN OVERSPECIFIED GAUSS POINT NUMBER (DEFAULT TO THREE)
C
      IF (NGP.LE.0.OR.NGP.GT.6) NGP=3                                   CUB0220
      IDGP=NGP-1                                                        CUB0230
C
CC    SELECT THE INTEGRATION POINTS AND CALCULATE THE WEIGHT FACTORS
C
      DO 300 I=1,NGP                                                    CUB0240
      XI=GP(I,IDGP)                                                     CUB0250
      WX=WF(I,IDGP)                                                     CUB0260
      DO 300 J=1,NGP                                                    CUB0270
      ETA=GP(J,IDGP)                                                    CUB0280
      WY=WF(J,IDGP)                                                     CUB0290
      DO 300 K=1,NGP                                                    CUB0300
      ZETA=GP(K,IDGP)                                                   CUB0310
      WZ=WF(K,IDGP)                                                     CUB0320
      WFACT=WX*WY*WZ                                                    CUB0330
C
CC    FORM THE ELEMENT PRODUCT MATRICES
C
      IF (ITYPE.EQ.1) GO TO 100                                         CUB0340
      CALL FORMC(XI,ETA,ZETA,NPEL)                                      CUB0350
      GO TO 200                                                         CUB0360
```

```
  100 CALL FORMH(XI,ETA,ZETA,NPEL)                                      CUB0370
  200 DO 250 L=1,NPEL                                                   CUB0380
      DO 250 M=1,NPEL                                                   CUB0390
  250 ENW(L,M)=ENW(L,M)+WFACT*ENN(L,M)                                  CUB0400
  300 CONTINUE                                                          CUB0410
      DO 350 I=1,NPEL                                                   CUB0420
      DO 350 J=I,NPEL                                                   CUB0430
      ENW(I,J)=5.D-1*(ENW(I,J)+ENW(J,I))                                CUB0440
  350 ENW(J,I)=ENW(I,J)                                                 CUB0450
      RETURN                                                            CUB0460
C
CC    THE FOLLOWING PERTAINS TO BOUNDARY CONDITION CALCULATIONS ONLY
C
      ENTRY CUBEB(NPEL,NGP,KBT,KBS)                                     CUB0470
      IF (NGP.LE.0.OR.NGP.GT.6) NGP=3                                   CUB0480
      IDGP=NGP-1                                                        CUB0490
      KBSABS=IABS(KBS)                                                  CUB0500
      GO TO (400,600,600),KBT                                           CUB0510
C
CC    VOLUME INTEGRAL BOUNDARY CONDITIONS
C
  400 DO 500 I=1,NGP                                                    CUB0520
      XI=GP(I,IDGP)                                                     CUB0530
      WX=WF(I,IDGP)                                                     CUB0540
      DO 500 J=1,NGP                                                    CUB0550
      ETA=GP(J,IDGP)                                                    CUB0560
      WY=WF(J,IDGP)                                                     CUB0570
      DO 500 K=1,NGP                                                    CUB0580
      ZETA=GP(K,IDGP)                                                   CUB0590
      WZ=WF(K,IDGP)                                                     CUB0600
      WFACT=WX*WY*WZ                                                    CUB0610
      CALL FORMV(XI,ETA,ZETA,NPEL)                                      CUB0620
      DO 450 L=1,NPEL                                                   CUB0630
  450 FE(L)=FE(L)-WFACT*COL(L)                                          CUB0640
  500 CONTINUE                                                          CUB0650
      GO TO 2000                                                        CUB0660
C
CC    SURFACE INTEGRAL BOUNDARY CONDITION
C
  600 ALF=0.0D0                                                         CUB0670
      C3=DFLOAT(KBS/KBSABS)                                             CUB0680
      DO 1500 I=1,NGP                                                   CUB0690
      C1=GP(I,IDGP)                                                     CUB0700
      W1=WF(I,IDGP)                                                     CUB0710
      DO 1500 J=1,NGP                                                   CUB0720
      C2=GP(J,IDGP)                                                     CUB0730
      W2=WF(J,IDGP)                                                     CUB0740
      WFACT=W1*W2                                                       CUB0750
```

```
      GO TO (700,800,900),KBSABS                                        CUB0760
  700 XI=C3                                                             CUB0770
      ETA=C1                                                            CUB0780
      ZETA=C2                                                           CUB0790
      GO TO 1000                                                        CUB0800
  800 XI=C1                                                             CUB0810
      ETA=C3                                                            CUB0820
      ZETA=C2                                                           CUB0830
      GO TO 1000                                                        CUB0840
  900 XI=C1                                                             CUB0850
      ETA=C2                                                            CUB0860
      ZETA=C3                                                           CUB0870
 1000 IF (KBT.EQ.3) GO TO 1200                                          CUB0880
      CALL FORMSQ(XI,ETA,ZETA,NPEL,KBSABS)                              CUB0890
      DO 1100 L=1,NPEL                                                  CUB0900
 1100 FE(L)=FE(L)+WFACT*COL(L)                                          CUB0910
      GO TO 1500                                                        CUB0920
 1200 CALL FORMSA(XI,ETA,ZETA,NPEL,KBSABS)                              CUB0930
      DO 1300 L=1,NPEL                                                  CUB0940
      DO 1300 M=1,NPEL                                                  CUB0950
 1300 ENW(L,M)=ENW(L,M)+WFACT*ENN(L,M)                                  CUB0960
 1500 CONTINUE                                                          CUB0970
      IF(KBT.NE.3) GO TO 2000                                           CUB0980
      DO 1600 I=1,NPEL                                                  CUB0990
      DO 1600 J=I,NPEL                                                  CUB1000
      ENW(I,J)=5.D-1*(ENW(I,J)+ENW(J,I))                                CUB1010
 1600 ENW(J,I)=ENW(I,J)                                                 CUB1020
 2000 RETURN                                                            CUB1030
      END                                                               CUB1040
```

```
      SUBROUTINE FORMC(XI,ETA,ZETA,NPEL)                                FOR0010
      IMPLICIT REAL*8 (A-H,O-Z)                                         FOR0020
      COMMON /MTRX/ CORD(32,3),ENN(32,32),DCOL(32,3),COL(32),AJ(3,3),AJI FOR0030
     1(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COARD(110 FOR0040
     2,3),F(110),PHI(110),PHIDOT(110),PHIWRK(110),PHIDDT(110),DTIM(5),EN FOR0050
     3DT(5),BGH(110,64),BGC(110,64)                                     FOR0051
      COMMON /FLP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM          FOR0060
      CALL SHAPE(XI,ETA,ZETA,NPEL)                                      FOR0070
      CALL JACOB(NPEL,AJ,DCOL,CORD,AJI,DTJ)                             FOR0080
C
C     SHAPE FUNCTION PRODUCTS
C
      DO 100 I=1,NPEL                                                   FOR0090
      A=COL(I)*DTJ                                                      FOR0100
      DO 100 J=1,NPEL                                                   FOR0110
  100 ENN(I,J)=A*COL(J)                                                 FOR0120
      RETURN                                                            FOR0130
      ENTRY FORMH(XI,ETA,ZETA,NPEL)                                     FOR0140
      CALL SHAPE(XI,ETA,ZETA,NPEL)                                      FOR0150
      CALL JACOB(NPEL,AJ,DCOL,CORD,AJI,DTJ)                             FOR0160
C
C     FORM THE SHAPE FUNCTION PARTIAL DERIVATIVES WITH RESPECT TO
C     GLOBAL COORDINATES
C
      DO 500 I=1,NPEL                                                   FOR0170
      DNX=DCOL(I,1)                                                     FOR0180
      DNY=DCOL(I,2)                                                     FOR0190
      DNZ=DCOL(I,3)                                                     FOR0200
      DCOL(I,1)=AJI(1,1)*DNX+AJI(1,2)*DNY+AJI(1,3)*DNZ                  FOR0210
      DCOL(I,2)=AJI(2,1)*DNX+AJI(2,2)*DNY+AJI(2,3)*DNZ                  FOR0220
  500 DCOL(I,3)=AJI(3,1)*DNX+AJI(3,2)*DNY+AJI(3,3)*DNZ                  FOR0230
C
C     PRODUCTS OF THE DERIVATIVES WITH REXPECT TO X
C
      DO 700 I=1,NPEL                                                   FOR0240
      BX=DCOL(I,1)*DTJ                                                  FOR0250
      DO 700 J=1,NPEL                                                   FOR0260
  700 ENN(I,J)=BX*AKX*DCOL(J,1)                                         FOR0270
C
C     PRODUCTS OF THE DERIVATIVES WITH REXPECT TO Y
C
      DO 800 I=1,NPEL                                                   FOR0280
      BY=DCOL(I,2)*DTJ                                                  FOR0290
      DO 800 J=1,NPEL                                                   FOR0300
  800 ENN(I,J)=BY*AKY*DCOL(J,2)+ENN(I,J)                                FOR0310
C
```

```
C       PRODUCTS OF THE DERIVATIVES WITH REXPECT TO Z
C
        DO 900 I=1,NPEL                                                 FOR0320
        BZ=DCOL(I,3)*DTJ                                                FOR0330
        DO 900 J=1,NPEL                                                 FOR0340
  900   ENN(I,J)=BZ*AKZ*DCOL(J,3)+ENN(I,J)                              FOR0350
        RETURN                                                          FOR0360
C
C       THE FOLLOWING PERTAINS TO BOUNDARY CONDITION CALCULATIONS ONLY
C
        ENTRY FORMV(XI,ETA,ZETA,NPEL)                                   FOR0370
        CALL SHAPE(XI,ETA,ZETA,NPEL)                                    FOR0380
        CALL JACOB(NPEL,AJ,DCOL,CORD,AJI,DTJ)                           FOR0390
        DO 1000 I=1,NPEL                                                FOR0400
 1000   COL(I)=DTJ*COL(I)                                               FOR0410
        RETURN                                                          FOR0420
        ENTRY FORMSQ(XI,ETA,ZETA,NPEL,KBSABS)                           FOR0430
        CALL SHAPE(XI,ETA,ZETA,NPEL)                                    FOR0440
        CALL JACOBB(NPEL,AJ,DCOL,CORD,DTJ,KBSABS)                       FOR0450
        DO 1100 I=1,NPEL                                                FOR0460
 1100   COL(I)=DTJ*COL(I)                                               FOR0470
        RETURN                                                          FOR0480
        ENTRY FORMSA(XI,ETA,ZETA,NPEL,KBSABS)                           FOR0490
        CALL SHAPE(XI,ETA,ZETA,NPEL)                                    FOR0500
        CALL JACOBB(NPEL,AJ,DCOL,CORD,DTJ,KBSABS)                       FOR0510
        DO 1200 I=1,NPEL                                                FOR0520
        A=COL(I)*DTJ                                                    FOR0530
        DO 1200 J=1,NPEL                                                FOR0540
 1200   ENN(I,J)=A*COL(J)                                               FOR0550
        RETURN                                                          FOR0560
        END                                                             FOR0570
```

```
      SUBROUTINE SHAPE(XI,ETA,ZETA,NPEL)                                SHP0010
      IMPLICIT REAL*8 (A-H,O-Z)                                         SHP0020
      COMMON /MTRX/ CORD(32,3),ENN(32,32),DCOL(32,3),COL(32),AJ(3,3),AJI SHP0030
     1(3,3),ENW(32,32),FE(32),PROPB(48),PROP(10,5),BGM(110,64),COARD(110 SHP0040
     2,3),F(110),PHI(110),PHIDOT(110),PHIWRK(110),PHIDDT(110),DTIM(5),EN SHP0050
     3DT(5),BGH(110,64),BGC(110,64)                                     SHP0051
      COMMON /FLP/ DTJ,AKX,AKY,AKZ,FACT,PHIREF,APZRO,ABIGN,TIM          SHP0060
      DIMENSION CORDX(4),CORDY(4),CORDZ(4),TRD(2)                       SHP0070
      DATA CORDX/1.0D0,-1.0D0,-1.0D0,1.0D0/                             SHP0080
      DATA CORDY/1.0D0,1.0D0,-1.0D0,-1.0D0/                             SHP0090
      DATA CORDZ/1.0D0,-1.0D0,1.0D0,-1.0D0/                             SHP0100
      DATA TRD/.33333333333333D0,-.33333333333333D0/                    SHP0110
C
      FL(XI,ETA,ZETA,X1,Y1,Z1)=(1.0D0+XI*X1)*(1.0D0+ETA*Y1)*(1.0D0+ZETA* SHP0120
     1Z1)/8.0D0                                                         SHP0130
      DFL(ETA,ZETA,X1,Y1,Z1)=X1*(1.0D0+ETA*Y1)*(1.0D0+ZETA*Z1)/8.0D0    SHP0140
C
      FQC(XI,ETA,ZETA,X1,Y1,Z1)=(1.0D0+XI*X1)*(1.0D0+ETA*Y1)*(1.0D0+ZETA SHP0150
     1*Z1)*(XI*X1+ETA*Y1+ZETA*Z1-2.0D0)/8.0D0                           SHP0160
      FQM(XI,ETA,ZETA,Y1,Z1)=(1.0D0-XI*XI)*(1.0D0+ETA*Y1)*(1.0D0+ZETA*Z1 SHP0170
     1)/4.0D0                                                           SHP0180
      DFQC(XI,ETA,ZETA,X1,Y1,Z1)=X1*(1.0D0+ETA*Y1)*(1.0D0+ZETA*Z1)*(2.0D SHP0190
     10*XI*X1+ETA*Y1+ZETA*Z1-1.0D0)/8.0D0                               SHP0200
      DFQMX(XI,ETA,ZETA,Y1,Z1)=-XI*(1.0D0+ETA*Y1)*(1.0D0+ZETA*Z1)/2.0D0 SHP0210
      DFQMY(XI,ZETA,Y1,Z1)=Y1*(1.0D0-XI*XI)*(1.0D0+ZETA*Z1)/4.0D0       SHP0220
C
      FCC(XI,ETA,ZETA,X1,Y1,Z1)=(1.0D0+XI*X1)*(1.0D0+ETA*Y1)*(1.0D0+ZETA SHP0230
     1*Z1)*(9.0D0*(XI*XI+ETA*ETA+ZETA*ZETA)-1.9D1)/6.4D1                SHP0240
      FCM(XI,ETA,ZETA,X1,Y1,Z1)=9.0D0*(1.0D0-XI*XI)*(1.0D0+9.0D0*XI*X1)* SHP0250
     1(1.0D0+ETA*Y1)*(1.0D0+ZETA*Z1)/6.4D1                              SHP0260
      DFCC(XI,ETA,ZETA,X1,Y1,Z1)=(1.0D0+ETA*Y1)*(1.0D0+ZETA*Z1)*(9.0D0*X SHP0270
     11*(XI*XI+ETA*ETA+ZETA*ZETA)-1.9D1*X1+1.8D1*XI+1.8D1*XI*XI*X1)/6.4D SHP0280
     21                                                                 SHP0290
      DFCMX(XI,ETA,ZETA,X1,Y1,Z1)=9.0D0*(1.0D0+ETA*Y1)*(1.0D0+ZETA*Z1)*( SHP0300
     1-2.0D0*XI-2.7D1*XI*XI*X1+9.0D0*X1)/6.4D1                          SHP0310
      DFCMY(XI,ZETA,X1,Y1,Z1)=9.0D0*(1.0D0-XI*XI)*(1.0D0+9.0D0*XI*X1)*Y1 SHP0320
     1*(1.0D0+ZETA*Z1)/6.4D1                                            SHP0330
C
      IGO=(NPEL-8)/12+1                                                 SHP0340
      GO TO (10,20,30),IGO                                              SHP0350
C
C     LINEAR ELEMENT FUNCTIONS
C
   10 ICOL=0                                                            SHP0360
      DO 100 IZ=1,2                                                     SHP0370
      DO 100 IX=1,4                                                     SHP0380
```

```
      ICOL=ICOL+1                                                        SHP0390
      X1=CORDX(IX)                                                       SHP0400
      Y1=CORDY(IX)                                                       SHP0410
      Z1=CORDZ(IZ)                                                       SHP0420
      COL(ICOL)=FL(XI,ETA,ZETA,X1,Y1,Z1)                                 SHP0430
      DCOL(ICOL,1)=DFL(ETA,ZETA,X1,Y1,Z1)                                SHP0440
      DCOL(ICOL,2)=DFL(ZETA,XI,Y1,Z1,X1)                                 SHP0450
  100 DCOL(ICOL,3)=DFL(XI,ETA,Z1,X1,Y1)                                  SHP0460
      GO TO 40                                                           SHP0470
C
CCC   QUADRATIC ELEMEMT FUNCTIONS
CCC
CCC
CCC   CORNER NODES (1,3,5,7,13,15,17,19)
C
   20 DO 200 IZ=1,2                                                      SHP0480
      II=12*(IZ-1)+1                                                     SHP0490
      IT=II+6                                                            SHP0500
      IX=0                                                               SHP0510
      DO 200 ICOL=II,IT,2                                                SHP0520
      IX=IX+1                                                            SHP0530
      Y1=CORDY(IX)                                                       SHP0540
      X1=CORDX(IX)                                                       SHP0550
      Z1=CORDZ(IZ)                                                       SHP0560
      COL(ICOL)=FQC(XI,ETA,ZETA,X1,Y1,Z1)                                SHP0570
      DCOL(ICOL,1)=DFQC(XI,ETA,ZETA,X1,Y1,Z1)                            SHP0580
      DCOL(ICOL,2)=DFQC(ETA,ZETA,XI,Y1,Z1,X1)                            SHP0590
  200 DCOL(ICOL,3)=DFQC(ZETA,XI,ETA,Z1,X1,Y1)                            SHP0600
C
CCC   MID SIDE NODES:  2, 6,14,18
CCC                    4, 8,16,20
CCC                    9,10,11,12
C
      IC9=8                                                              SHP0610
      DO 250 IZ=1,2                                                      SHP0620
      II=12*(IZ-1)+2                                                     SHP0630
      IT=II+4                                                            SHP0640
      DO 250 IC2=II,IT,4                                                 SHP0650
      IC4=IC2+2                                                          SHP0660
      IC9=IC9+1                                                          SHP0670
      IN=IC9-8                                                           SHP0680
      X1=CORDX(IN)                                                       SHP0690
      Y1=CORDY(IN)                                                       SHP0700
      Z1=CORDZ(IN)                                                       SHP0710
      COL(IC2)=FQM(XI,ETA,ZETA,Z1,Y1)                                    SHP0720
      COL(IC4)=FQM(ETA,ZETA,XI,Y1,-Z1)                                   SHP0730
      COL(IC9)=FQM(ZETA,XI,ETA,X1,Y1)                                    SHP0740
      DCOL(IC2,1)=DFQMX(XI,ETA,ZETA,Z1,Y1)                               SHP0750
```

```
      DCOL(IC2,2)=DFQMY(XI,ZETA,Z1,Y1)                                  SHP0760
      DCOL(IC2,3)=DFQMY(XI,ETA,Y1,Z1)                                   SHP0770
      DCOL(IC4,1)=DFQMY(ETA,ZETA,-Z1,Y1)                                SHP0780
      DCOL(IC4,2)=DFQMX(ETA,XI,ZETA,-Z1,Y1)                             SHP0790
      DCOL(IC4,3)=DFQMY(ETA,XI,Y1,-Z1)                                  SHP0800
      DCOL(IC9,1)=DFQMY(ZETA,ETA,X1,Y1)                                 SHP0810
      DCOL(IC9,2)=DFQMY(ZETA,XI,Y1,X1)                                  SHP0820
  250 DCOL(IC9,3)=DFQMX(ZETA,XI,ETA,X1,Y1)                              SHP0830
      GO TO 40                                                          SHP0840
C
CC    CUBIC ELEMENT FUNCTIONS
CCC
CC    CORNER NODES (1,4,7,10/21,24,27,30)
C
   30 DO 300 IZ=1,2                                                     SHP0850
      II=20*(IZ-1)+1                                                    SHP0860
      IT=II+9                                                           SHP0870
      IX=0                                                              SHP0880
      DO 300 ICOL=II,IT,3                                               SHP0890
      IX=IX+1                                                           SHP0900
      X1=CORDX(IX)                                                      SHP0910
      Y1=CORDY(IX)                                                      SHP0920
      Z1=CORDZ(IZ)                                                      SHP0930
      COL(ICOL)=FCC(XI,ETA,ZETA,X1,Y1,Z1)                               SHP0940
      DCOL(ICOL,1)=DFCC(XI,ETA,ZETA,X1,Y1,Z1)                           SHP0950
      DCOL(ICOL,2)=DFCC(ETA,ZETA,XI,Y1,Z1,X1)                           SHP0960
  300 DCOL(ICOL,3)=DFCC(ZETA,XI,ETA,Z1,X1,Y1)                           SHP0970
C
CC    MID SIDE NODES:  2, 9,22,29 //  3, 8,23,28
CC                     5,12,25,32 // 6,11,26,31
CC                    13,14,15,16, // 17,18,19,20
C
      I13=12                                                            SHP0980
      DO 350 IZ=1,2                                                     SHP0990
      ID=9-2*IZ                                                         SHP1000
      IX=0                                                              SHP1010
      DO 350 II=1,2                                                     SHP1020
      IS=20*II-19+IZ                                                    SHP1030
      IT=20*II-10-IZ                                                    SHP1040
      DO 350 ICOL=IS,IT,ID                                              SHP1050
      I13=I13+1                                                         SHP1060
      I5=ICOL+3                                                         SHP1070
      IX=IX+1                                                           SHP1080
      X1=CORDX(IX)                                                      SHP1090
      Y1=CORDY(IX)                                                      SHP1100
      Z1=CORDZ(IX)                                                      SHP1110
      THIRD=TRD(IZ)                                                     SHP1120
```

```
      COL(ICOL)=FCM(XI,ETA,ZETA,THIRD,Z1,Y1)                                    SHP1130
      COL(I5)=FCM(ETA,ZETA,XI,THIRD,Y1,-Z1)                                     SHP1140
      COL(I13)=FCM(ZETA,XI,ETA,THIRD,X1,Y1)                                     SHP1150
      DCOL(ICOL,1)=DFCMX(XI,ETA,ZETA,THIRD,Z1,Y1)                               SHP1160
      DCOL(ICOL,2)=DFCMY(XI,ZETA,THIRD,Z1,Y1)                                   SHP1170
      DCOL(ICOL,3)=DFCMY(XI,ETA,THIRD,Y1,Z1)                                    SHP1180
      DCOL(I5,1)=DFCMY(ETA,ZETA,THIRD,-Z1,Y1)                                   SHP1190
      DCOL(I5,2)=DFCMX(ETA,ZETA,XI,THIRD,Y1,-Z1)                                SHP1200
      DCOL(I5,3)=DFCMY(ETA,XI,THIRD,Y1,-Z1)                                     SHP1210
      DCOL(I13,1)=DFCMY(ZETA,ETA,THIRD,X1,Y1)                                   SHP1220
      DCOL(I13,2)=DFCMY(ZETA,XI,THIRD,Y1,X1)                                    SHP1230
  350 DCOL(I13,3)=DFCMX(ZETA,XI,ETA,THIRD,X1,Y1)                                SHP1240
   40 RETURN                                                                    SHP1250
      END                                                                       SHP1260
```

```
      SUBROUTINE JACOB (NPEL,AJ,DCOL,CORD,AJI,DTJ)                      JAC0010
C
C     THIS SUBROUTINE WILL FORM THE JACOBIAN AND COMPUTE ITS DETERMINANT
C     AND INVERSE.
C             AJ= 3 X 3 JACOBIAN MATRIX
C             AJI= 3 X 3 INVERSE OF THE JACOBIAN
C             DTJ=DETERMINANT OF THE JACOBIAN
C             CORD= NPEL X 3 MATRIX OF GLOBAL COORDINATES
C             DCOL= NPEL X 3 MATRIX OF PARTIAL DERIVATIVES WITH RESPECT
C                   TO LOCAL COORDINATES
C             NPEL= NUMBER OF NODES PER ELEMENT
C
      IMPLICIT REAL*8 (A-H,O-Z)                                         JAC0020
      DIMENSION AJ(3,3),AJI(3,3),CORD(32,3),DCOL(32,3)                  JAC0030
C
C     FORM THE JACOBIAN
C
      DO 500 I=1,3                                                      JAC0040
      DO 500 J=1,3                                                      JAC0050
      AJ(I,J)=0.0D0                                                     JAC0060
      DO 500 K=1,NPEL                                                   JAC0070
  500 AJ(I,J)=AJ(I,J)+DCOL(K,I)*CORD(K,J)                               JAC0080
C
C     COMPUTE THE DETERMINANT
C
      DTJ=AJ(1,1)*AJ(2,2)*AJ(3,3)+AJ(1,2)*AJ(2,3)*AJ(3,1)+AJ(1,3)*AJ(2,1 JAC0090
     1)*AJ(3,2)-AJ(3,1)*AJ(2,2)*AJ(1,3)-AJ(3,2)*AJ(2,3)*AJ(1,1)-AJ(3,3)* JAC0100
     2AJ(2,1)*AJ(1,2)                                                   JAC0110
C
C     COMPUTE THE INVERSE
C
      AJI(1,1)= (AJ(2,2)*AJ(3,3)-AJ(3,2)*AJ(2,3))/DTJ                   JAC0120
      AJI(2,1)=-(AJ(2,1)*AJ(3,3)-AJ(3,1)*AJ(2,3))/DTJ                   JAC0130
      AJI(3,1)= (AJ(2,1)*AJ(3,2)-AJ(3,1)*AJ(2,2))/DTJ                   JAC0140
      AJI(1,2)=-(AJ(1,2)*AJ(3,3)-AJ(3,2)*AJ(1,3))/DTJ                   JAC0150
      AJI(2,2)= (AJ(1,1)*AJ(3,3)-AJ(3,1)*AJ(1,3))/DTJ                   JAC0160
      AJI(3,2)=-(AJ(1,1)*AJ(3,2)-AJ(3,1)*AJ(1,2))/DTJ                   JAC0170
      AJI(1,3)= (AJ(1,2)*AJ(2,3)-AJ(2,2)*AJ(1,3))/DTJ                   JAC0180
      AJI(2,3)=-(AJ(1,1)*AJ(2,3)-AJ(2,1)*AJ(1,3))/DTJ                   JAC0190
      AJI(3,3)= (AJ(1,1)*AJ(2,2)-AJ(2,1)*AJ(1,2))/DTJ                   JAC0200
      RETURN                                                            JAC0210
C
C     THE FOLLOWING PERTAINS TO BOUNDARY CONDITION CALCULATIONS ONLY
C
      ENTRY JACOBB(NPEL,AJ,DCOL,CORD,DTJ,KS)                            JAC0220
C
```

```
C       FORM THE JACOBIAN
C
        DO 600 I=1,3                                                    JAC0230
        DO 600 J=1,3                                                    JAC0240
        AJ(I,J)=0.0D0                                                   JAC0250
        DO 600 K=1,NPEL                                                 JAC0260
  600   AJ(I,J)=AJ(I,J)+DCOL(K,I)*CORD(K,J)                            JAC0270
C
C       COMPUTE THE SURFACE DETERMINANT
C
        GO TO (700,800,900),KS                                          JAC0280
  700   DTJ=AJ(2,2)*AJ(3,3)-AJ(2,3)*AJ(3,2)                            JAC0290
        GO TO 1000                                                      JAC0300
  800   DTJ=AJ(1,1)*AJ(3,3)-AJ(3,1)*AJ(1,3)                            JAC0310
        GO TO 1000                                                      JAC0320
  900   DTJ=AJ(1,1)*AJ(2,2)-AJ(2,1)*AJ(1,2)                            JAC0330
 1000   RETURN                                                          JAC0340
        END                                                             JAC0350
```

```
      SUBROUTINE LDLT(A,N,M,ASN)                                        LDS0010
C*************************************************************
C    THIS SUBROUTINE DECOMPOSES THE COEFFICIENT MATRIX A* OF THE LINEAR
C    BANDED SYMMETRICAL SYSTEM (A*)(X) = (B), INTO THE PRODUCT (L)(D)(LT)
C    THE UPPER BAND ONLY IS USED, AND IT MUST BE STORED IN A RECTANGULAR
C    ARRAY (A) WITH ALL ITS DIAGONAL ELEMENTS IN THE FIRST COLUMN
C    MATRIX (A) IS DESTROYED IN THE PROCESS AND THE RESULTS RETURNED IN
C    THE SAME ARRAY. MATRIX (D) IS PUT IN THE FIRST COLUMN AND THE REST
C    OF THE MATRIX CONTAINS THE ELEMENTS OF THE UNIT TRIANGULAR MATRIX (LT)
C    IN BANDED FORM WITHOUT ITS IMPLIED UNIT DIAGONAL ELEMENTS.
C
C         A IS THE NAME OF THE COEFFICIENT MATRIX
C         N IS THE NUMBER OF EQUATIONS IN THE SYSTEM
C         M IS THE HALF BAND WIDTH OF THE SYSTEM
C         ASN IS THE CALCULATED VALUE OF ZERO
C
C    CODED BY GILLES CANTIN, NAVAL POSTGRADUATE SCHOOL, MAY 1972
C*************************************************************
      IMPLICIT REAL*8(A-H,O-Z)                                          LDS0019
      DIMENSION A(110,64)                                               LDS0020
      AVSN=ASN*ASN                                                      LDS0030
      NM=N-1                                                            LDS0040
      DO 300 I=1,NM                                                     LDS0050
      DIAG=A(I,1)                                                       LDS0060
      DO 100 J=2,M                                                      LDS0070
      IF(DABS(A(I,J)).LE.AVSN) A(I,J)=0.0D0                             LDS0080
      A(I,J)=A(I,J)/DIAG                                                LDS0090
  100 CONTINUE                                                          LDS0100
      DO 300 J=2,M                                                      LDS0110
      L=I+J-1                                                           LDS0120
      IF(L.GT.N) GO TO 300                                              LDS0130
      IF(DABS(A(I,J)).LE.AVSN) GO TO 300                                LDS0140
      AA=A(I,J)*DIAG                                                    LDS0150
      DO 200 K=J,M                                                      LDS0160
      ML=1+K-J                                                          LDS0170
  200 A(L,ML)=A(L,ML)-AA*A(I,K)                                         LDS0180
  300 CONTINUE                                                          LDS0190
      RETURN                                                            LDS0200
      ENTRY SLV(Z,B,N,M)                                                LDS0210
C*************************************************************
C    THIS SUBROUTINE PERFORMS A FORWARD SUBSTITUTION, FOLLOWED BY A
C    BACKWARD SUBSTITUTION FOR THE SYSTEM (A*)(X) = (B), AFTER THE
C    MATRIX (A) HAS BEEN DECOMPOSED BY THE SUBROUTINE LDLT
C         A  IS THE NAME OF THE DECOMPOSED MATRIX OF COEFFICIENTS
C         B  IS THE NAME OF THE COLUMN LOAD VECTOR OR RIGHT HAND MEMBER
C            THE ANSWERS ARE RETURNED IN THIS VECTOR UPON EXIT
```

```
C         N  IS THE NUMBER OF EQUATIONS IN THE SYSTEM
C         M  IS THE HALF BAND WIDTH OF THE SYSTEM
C
C    AFTER ONE CALL TO LDLT, ONE CALL OF SLV PER LOAD VECTOR IS REQUIRED
C
C    CODED BY GILLES CANTIN, NAVAL POSTGRADUATE SCHOOL, MAY 1972
C ***************************************************************
      DIMENSION Z(110,64),B(110)                                    LDS0220
      NM=N-1                                                        LDS0230
      DO 400 I=1,NM                                                 LDS0240
      BI=B(I)                                                       LDS0250
      DO 400 J=2,M                                                  LDS0260
      L=I+J-1                                                       LDS0270
      IF (L.GT.N) GO TO 400                                         LDS0280
      B(L)=B(L)-Z(I,J)*BI                                           LDS0290
  400 CONTINUE                                                      LDS0300
      DO 500 I=1,N                                                  LDS0310
  500 B(I)=B(I)/Z(I,1)                                              LDS0320
      DO 600 L=2,N                                                  LDS0330
      IR=N-L+1                                                      LDS0340
      BIRP=B(IR+1)                                                  LDS0350
      DO 600 J=2,M                                                  LDS0360
      IRO=IR-J+2                                                    LDS0370
      IF(IRO.LE.0) GO TO 600                                        LDS0380
      B(IRO)=B(IRO)-Z(IRO,J)*BIRP                                   LDS0390
  600 CONTINUE                                                      LDS0400
      RETURN                                                        LDS0410
      END                                                           LDS0420
```

```
      SUBROUTINE MARK(C,K,M)                                                    MAR0010
      IMPLICIT REAL*8(A-H,O-Z)                                                  MAR0020
      DIMENSION C(110,3)                                                        MAR0030
      PI=3.14159265358979300                                                    MAR0040
      IF(K.NE.2) GO TO 300                                                      MAR0050
      DO 100 I=1,M                                                              MAR0060
      ANG=C(I,2)*PI/1.8D2                                                       MAR0070
      R=C(I,1)                                                                  MAR0080
      C(I,1)=R*DCOS(ANG)                                                        MAR0090
  100 C(I,2)=R*DSIN(ANG)                                                        MAR0100
      WRITE(6,200)                                                              MAR0110
  200 FORMAT('0',5X,'COORDINATES GIVEN IN CYLINDRICAL SYSTEM OF REFERENC        MAR0120
     1E',//,6X,'RECTANGULAR COORDINATES FOLLOW',///,5X,' NODE NUMBER',10        MAR0130
     2X,'X COORDINATE',12X,'Y COORDINATE',12X,'Z COORDINATE',///)                MAR0140
      GO TO 600                                                                 MAR0150
  300 DO 400 I=1,M                                                              MAR0160
      ANGZ=C(I,3)*PI/1.8D2                                                      MAR0170
      ANGX=C(I,2)*PI/1.8D2                                                      MAR0180
      R=C(I,1)                                                                  MAR0190
      SIN=DSIN(ANGZ)                                                            MAR0200
      C(I,1)=R*SIN*DCOS(ANGX)                                                   MAR0210
      C(I,2)=R*SIN*DSIN(ANGX)                                                   MAR0220
  400 C(I,3)=R*DCOS(ANGZ)                                                       MAR0230
      WRITE(6,500)                                                              MAR0240
  500 FORMAT('0',5X,'COORDINATES GIVEN IN SPHERICAL SYSTEM OF REFERENCE'        MAR0250
     1,//,6X,'RECTANGULAR COORDINATES FOLLOW',///,5X,' NODE NUMBER',10X,        MAR0260
     2'X COORDINATE',12X,'Y COORDINATE',12X,'Z COORDINATE',///)                 MAR0270
  600 DO 700 J=1,M                                                              MAR0280
  700 WRITE(6,800) J,(C(J,L),L=1,3)                                             MAR0290
  800 FORMAT(10X,I5,5X,1P3G24.16)                                               MAR0300
      RETURN                                                                    MAR0310
      END                                                                       MAR0320
```

```
      SUBROUTINE BNSN(A,N,APZRO,ABIGN)                                  BSN0010
      IMPLICIT REAL*8(A-H,O-Z)                                          BSN0020
      DIMENSION A(110,64)                                               BSN0030
      ANB=0.0D0                                                         BSN0040
      DO 100 I=1,N                                                      BSN0050
  100 ANB=ANB+DABS(A(I,1))                                              BSN0060
      APZRO=ANB*1.0D-15/DFLOAT(N)                                       BSN0070
      ABIGN=APZRO*1.0D30                                                BSN0080
      RETURN                                                            BSN0090
      END                                                               BSN0100

      SUBROUTINE MULT(A,B,C,N,M)                                        MUT0010
      IMPLICIT REAL*8(A-H,O-Z)                                          MUT0020
      DIMENSION A(110,64),B(110),C(110)                                 MUT0030
      DO 200 I=1,N                                                      MUT0040
      IST=I-M-1                                                         MUT0050
      C(I)=B(I)*A(I,1)                                                  MUT0060
      DO 200 J=2,M                                                      MUT0070
      IROW=IST+J                                                        MUT0080
      IF(IROW.LE.0) GO TO 100                                           MUT0090
      ICOL=M-J+2                                                        MUT0100
      C(I)=C(I)+A(IROW,ICOL)*B(IROW)                                    MUT0110
  100 IR=I+J-1                                                          MUT0120
      IF(IR.GT.N) GO TO 200                                             MUT0130
      C(I)=C(I)+A(I,J)*B(IR)                                            MUT0140
  200 CONTINUE                                                          MUT0150
      RETURN                                                            MUT0160
      END                                                               MUT0170
```

```
      SUBROUTINE STJFF(A,B,N,M)                                          STF0010
C
CCC   THIS SUBROUTINE WILL PLACE THE CONTENTS OF MATRIX "A" INTO
CCC   MATRIX "B"; MATRIX "A" IS PRESERVED; N = NUMBER OF ROWS;
CCC   M = NUMBER OF COLUMNS
C
      IMPLICIT REAL*8(A-H,O-Z)                                           STF0020
      DIMENSION A(110,64),B(110,64)                                      STF0030
      DO 100 I=1,N                                                       STF0040
      DO 100J=1,M                                                        STF0050
  100 B(I,J)=A(I,J)                                                      STF0060
      RETURN                                                             STF0070
      ENTRY STUFFV(AA,BB,N,MC)                                           STF0080
C
CCC   PLACES VECTOR "BB" INTO SPECIFIED COLUMN (MC) OF MATRIX "AA"
C
      DIMENSION AA(110,3),BB(110)                                        STF0090
      DO 200 I=1,N                                                       STF0100
  200 AA(I,MC)=BB(I)                                                     STF0110
      RETURN                                                             STF0120
      END                                                                STF0130


      SUBROUTINE PUNC(A,B,N,K)                                           PUN0010
      IMPLICIT REAL*8(A-H,O-Z)                                           PUN0020
      DIMENSION A(110),B(110)                                            PUN0030
      IF (K.GT.2) GO TO 300                                              PUN0040
      DO 100 I=1,N                                                       PUN0050
  100 WRITE(7,200) I,A(I)                                                PUN0060
  200 FORMAT(I10,D24.16)                                                 PUN0070
      GO TO 600                                                          PUN0080
  300 DO 400 I=1,N                                                       PUN0090
  400 WRITE(7,500) I,A(I),B(I)                                           PUN0100
  500 FORMAT(I10,2D24.16)                                                PUN0110
  600 RETURN                                                             PUN0120
      END                                                                PUN0130
```

LIST OF REFERENCES

1. Arpaci, V. S., Conduction Heat Transfer, Addison-Wesley Publishing Company, 1966.

2. Berg, P. W. and McGregor, J. L., Elementary Partial Differential Equations, Holden-Day, 1966.

3. Crandall, S. H., Engineering Analysis, McGraw-Hill, 1956.

4. Haji-Sheikh, A. and Sparrow, E. M., "Transient Heat Conduction in a Prolate Spheroidal Solid," Tran. ASMEHT, v. 88, p. 331-3, August 1966.

5. Holman, J. P., Heat Transfer, 2d ed., Mc-Graw Hill, 1963.

6. Irons, B. M., "A Frontal Solution Program for Finite Element Analysis," International Journal for Numerical Methods in Engineering, v. 2, p. 5-32, 1970.

7. Johnson, D. E., "A Proof of the Stability of the Houbolt Method," AIAA Journal, v. 4, No. 7, p. 1450-1451, August 1966.

8. Newmark, N. M., "A Method of Computation for Structural Dynamics," ASCE, Journal of Engineering Mechanics Division, v. 85, No. EM3, p. 67-64, July 1959.

9. Stroud, A. H. and Secrest, D., Gaussian Quadrature Formulas, Prentice-Hall Inc., 1966.

10. Zienkiewicz, O. C., The Finite Element Method in Engineering Science, McGraw-Hill, 1971.

11. Zienkiewicz, O. C. and Parekh, C. J., "Transient Field Problems: Two-Dimensional and Three-Dimensional Analysis by Isoparametric Finite Elements," International Journal for Numerical Methods in Engineering, v. 2, No. 1, p. 61-71, January-March 1970.

12. Users Manual, W. R. Church Computer Center, U.S. Naval Postgraduate School, Monterey, California, 1970.

13. Private communication from Gilles Cantin.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Professor Gilles Cantin, Code 59Ci Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	5
4.	Asst Professor David Salinas, Code 59Zc Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	5
5.	LCDR Girard T. Lew Supervisor of Shipbuilding, Conversion, and Repair, USN Pascagoula, Mississippi 39567	3
6.	Professor Paul F. Pucci, Code 59Pc Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
7.	Professor Robert E. Newton, Code 59Ne Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
8.	Professor John E. Brock, Code 59Bc Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
9.	Dr. Arnold Adicoff Naval Weapons Center China Lake, California 93555	5
10.	W. R. Church Computer Center, 0211 Naval Postgraduate School Monterey, California 93940	2

DOCUMENT CONTROL DATA - R & D

ecurity classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

TING ACTIVITY (Corporate author)

val Postgraduate School
nterey, California 93940

2a. REPORT SECURITY CLASSIFICATION
Unclassified

2b. GROUP

TITLE

THREE DIMENSIONAL SOLUTION OF THE TRANSIENT FIELD PROBLEM
ING ISOPARAMETRIC FINITE ELEMENTS

PTIVE NOTES (Type of report and, inclusive dates)

ster's Thesis; December 1972

RS) (First name, middle initial, last name)

rard T. Lew; Lieutenant Commander, United States Navy

DATE	7a. TOTAL NO. OF PAGES	7b. NO OF REFS
cember 1972	122	13

RACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
CT NO.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

TRIBUTION STATEMENT

pproved for public release; distribution unlimited.

EMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Naval Postgraduate School Monterey, California 93940

STRACT

An isoparametric finite element formulation solving a eneral form of the transient field equation is presented in his thesis. The formulation is developed for fields in hree-dimensional Euclidean space. A FORTRAN IV computer rogram employing double precision arithmetic, compact storage echniques, and providing the option of several numerical inte- ration methods and types of finite elements is presented. ata input may be in rectangular, cylindrical, or spherical oordinates and variations in time integration step size are ermitted. Time dependent boundary conditions are not con- idered.

Comparisons of theoretical and computer solutions for a ariety of test problems demonstrate close agreement. Instruc- ions for use of the program are included.

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
FINITE ELEMENT						
FIELD						
TRANSIENT FIELD						
ISOPARAMETRIC FINITE ELEMENT						

Thesis
L604
c.1

Lew

A three dimensional solution of the transient field problem using isoparametric finite elements.

Thesis
L604
c.1

141635

Lew

A three dimensional solution of the transient field problem using isoparametric finite elements.